Web Application Development II

Week 9: Vue.js Event Handling

Agenda

Computed Property

Binding Form Inputs

List Rendering

Asynchronous requests with Vue (Axios)

Building a reactive frontend app

Computed Properties

Some properties may be derived from (dependent on) other (native) properties.

In this case, it is better to use computed properties.

A computed property is a function which is used as if it is a variable.

Example

wk9example1.html

Computed vs Methods

Computed

A computed property can be achieved by invoking a method.

<p>Full Name: "{{ getFullName() }}"</p>

Computed properties are cached based on their reactive dependencies.

A computed property only re-evaluates when some of its reactive dependencies have changed.

Methods

A method invocation always runs the function whenever a re-render happens.

Binding from Inputs

Use v-model to create two-way data bindings on form input, textarea, and select elements.

v-model uses different properties and emits different events for different input elements.

- text and textarea elements use value property and input event;
- checkboxes and radio buttons use checked property and change event;
- select fields use value as a prop and change as an event.

Example

wk9example2.html

Binding to Vue’s Dynamic Property

For radio, checkbox and select options, the v-model binding values are usually static strings (or booleans for checkbox).

Sometimes we may want to bind the value to a dynamic property on the Vue instance.

We can combine v-model with v-bind to achieve that:

Example

wk9example3.html

v-model Modifiers (non-examinable)

v-model can be associated with different modifiers which helps to pre-process the inputs.

Example:

```
<!-- synced after "change" instead of "input" -->
<input v-model.lazy="msg">

<!-- user input to be automatically typecast as a Number -->
<input v-model.number="age" type="number">

<!-- whitespace from user input to be trimmed -->
<input v-model.trim="msg">
```

v-model vs v-bind

v-bind

v-bind is used mainly for binding HTML attributes, e.g., src, class, style, and value.

Example

<a v-bind:href="url"> ... </a>

// is the same as

<a href=”{{ url }}”> ... </a>

v-model

v-model is used mainly for input and form binding; use it when dealing with various input types. It is a two-way binding for form inputs.

Example

<input v-model="something">

// is the same as

```
<input v-bind:value="something"
  v-on:input="something = event.target.value" >
```

Exercise 1

Given ex1-age.html, complete the script part by creating a Vue instance such that once a date of birth is entered, the age is displayed. Your code should be limited to the <script> part.

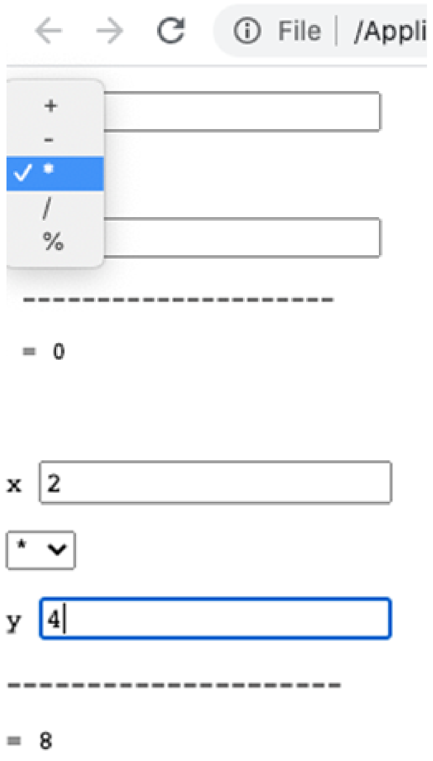
Hint: use computed properties

Hint: the following functions will be helpful: Date(), getFullYear(), getMonth(), getDate()

Exercise 2

Given ex2-calculator.html, complete the TODO so that it produces a web page that accepts three inputs: x and an operator (one of the 5) and y, and display the result accordingly.

Hint: Use computed property.



List Rendering with v-for

Use v-for directive to render a list of items based on an array.

The v-for directive supports a special syntax in the form of “item in items” (or “item of items”), where items is the source data array and item is an alias for the array element being iterated on.

v-for also supports an optional second argument for the index of the current item.

Example

wk9example4.html

v-for

v-for can also take an integer, in which case it will repeat the template that many times.

v-for can take an array or object.

Example

wk9example5.html

wk9example6.html

Note: When v-if and v-for are used together in the same HTML node, v-if has a higher priority than v-for. Not recommended to use them together. See below example:

```
<!-- This will throw an error because v-if is evaluated first
     and property ”student" is not defined yet -->

<li v-for=”student in students" v-if=”student.isInDeanList"> {{ student.name }} </li>
```

Rendering a Table

v-for can be easily used to render tables.

Example

wk9example7.html

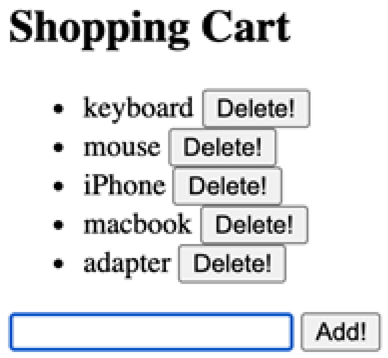
Take note of the clear separation of data and presentation.

Exercise 3

Given ex3-shoppingcart.html, complete the TODO so that it implements the following functionalities.

- When clicking the 'Delete' button, the corresponding item is removed from the list.
- When 'Add!' button is clicked, the item entered in the text box is added to the list.

Hint: use v-model, v-for



Compare this version with the version we did in Week 5.

Asynchronous Requests

Vue itself doesn’t contain any library to make Async. requests

However, it can work with existing, external libraries such as AXIOS for making Async. requests.

AXIOS is the recommended library for Vue for making HTTP requests.

And we have used AXIOS with JavaScript. It’s the same.

Example: A Blog Service

Set up a blog service on your WAMP server.

1. Start your WAMP server
2. Import create.sql
3. Modify the password and port number accordingly in ConnectionManagement.php
4. Put the folder blog on your server

Username: “root”;
password is “” for windows; “root” for MAC);
Default port is: 3308 (or 3306)

Test Case

http://localhost/is216/REST/blog/getPosts.php

Open the above (customize the URL according to where the blog folder is on your WAMP server) and make sure some JSON data is received.

Example: a Blog Service

- Sending AJAX call to getPosts.php using Vue and AXIOS.
- Display the post data in the form of a table using directives such as v-for and {{ }}

Example

wk9example8.html

Passing Data by GET

To pass data via GET requests, simply attach the data at the end of URL.

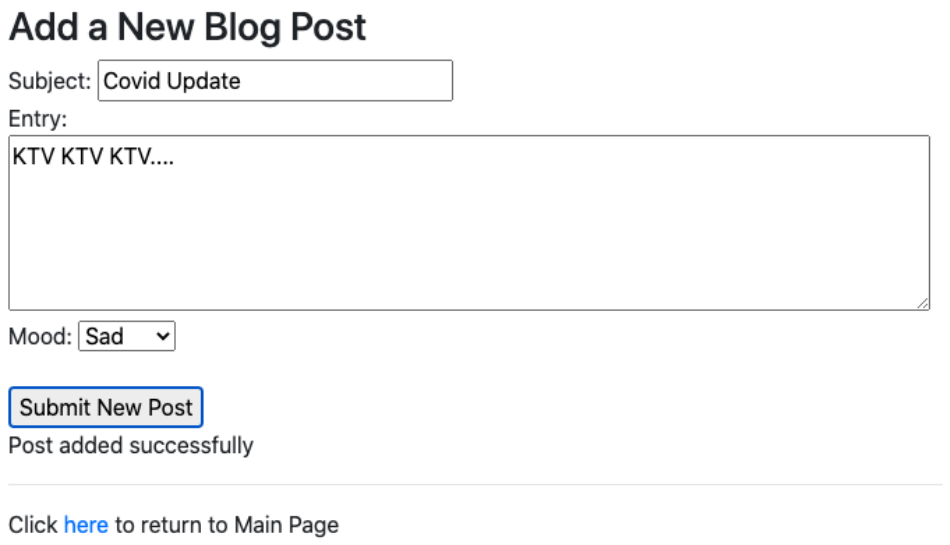
Example

```
Vue.createApp({
data() {
     return { entry: 'newPost’ }
},
created: function() {
  axios.get('https://localhost/is216/REST/blog/addPost.php’, {
                  params : { entry: this.entry }
   })
  .then(response => { console.log(response.data) })
  .catch(error => { console.log(error.message) })
}.mount(‘#app’)
```

Exercise 4

Setup the server as instructed “Example: A Blog Service”

Complete the TODO in ex4-addpost.html so that it uses the service provided by addPost.php to add a new blog post.



Add a New Blog Post

Subject: Covid Update

Entry:

KTV KTV KTV....

Mood: Sad

Post added successfully

Click here to return to Main Page

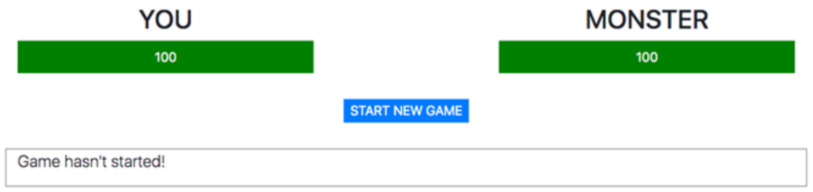
Putting Them Together

Build a Monster Slayer Game

Build a frontend game app that has the interface shown on the right.

Initially, both you and monster are given the health "100" as indicated in the green health bar.

The game status is displayed as "Game hasn't started!"

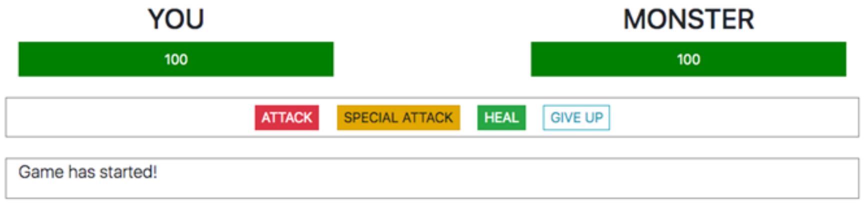


Game hasn't started!

Build a Monster Slayer Game

When you click on “START NEW GAME”, the game interface changes.

- The status changes to “Game has started!”
- Four buttons appear as shown above



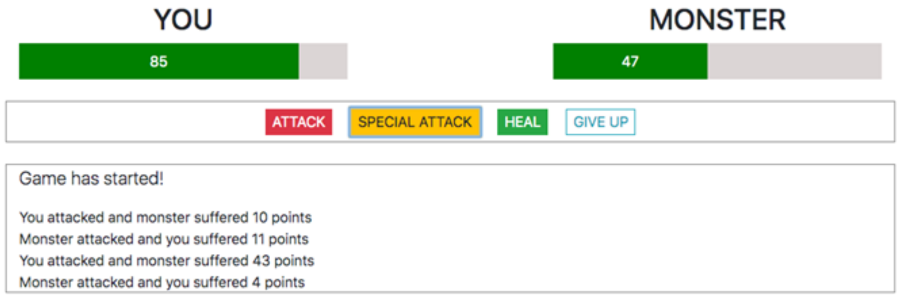
Game has started!

Build a Monster Slayer Game

When you "ATTACK", the monster suffers some (random) damage and its health bar reduces accordingly. When you "SPECIAL ATTACK", the monster would generally suffer more damage than usual.

At the same time, the monster attacks back and you suffers some damage and your health bar reduces accordingly.

The game status should display the messages as shown above.



Game has started!

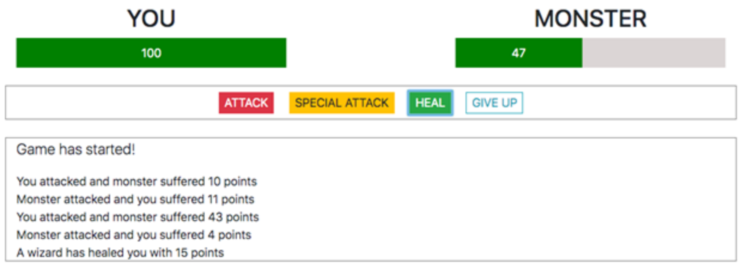
You attacked and monster suffered 10 points
Monster attacked and you suffered 11 points
You attacked and monster suffered 43 points
Monster attacked and you suffered 4 points

Build a Monster Slayer Game

“HEAL”: you will be (randomly) awarded with some health and your health bar increases accordingly:

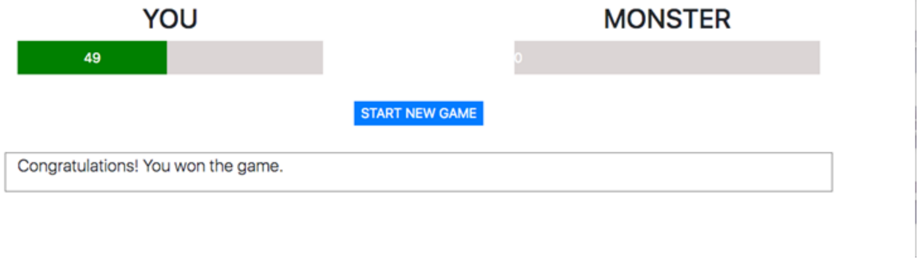
When one’s health is zero, one wins the game. “START NEW GAME” button appears again. You can re-start the game.

When you are losing, you can “GIVE UP”.

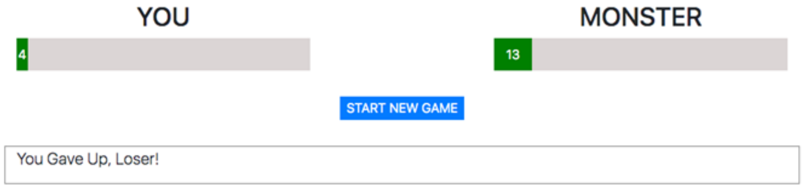


Game has started!

You attacked and monster suffered 10 points
Monster attacked and you suffered 11 points
You attacked and monster suffered 43 points
Monster attacked and you suffered 4 points
A wizard has healed you with 15 points



Congratulations! You won the game.



You Gave Up, Loser!

Build a Monster Slayer Game

Hints:

- Use Vue.js and never use document.getElement...
- Control the health bar size by: v-bind:style
- User v-if to control what buttons are to be displayed

Take Away Message

Use “computed” whenever possible.

Take note of all inputs are stored with v-model.

Use v-for to render structured data.