

# Vue Fundamentals - 03

## V-model and lifecycle hooks



Peter Kassenaar –  
[info@kassenaar.com](mailto:info@kassenaar.com)



# Using v-model

Two-way databinding with Vue

## Using v-model to select changes

*"You can use the `v-model` directive to create **two-way data bindings** on form input, textarea, and select elements. It automatically picks the correct way to update the element based on the input type."*

## Using `v-model`

### *Two-way data binding*

Reflect changes in the UI back to the component  
and the other way around

```
<input type="text" v-model="...">
```


# Push items to (new) array

**New Countries**

Add Country

Canada

France



```
<input type="text" class="form-control-lg"  
      v-model="newCountry"  
      @keyup.enter="addCountry()"  
>  
<button @click="addCountry()" class="btn btn-info">  
  Add Country  
</button>
```

```
methods: {  
  selectCountry(index) {  
    this.selectedCountryIndex = index;  
  },  
  addCountry(){  
    this.newCountries.push(this.newCountry);  
    this.newCountry='';  
  },  
},  
computed: {
```

## Using `v-model` on check boxes

```
<input type="checkbox" v-model="...">
```

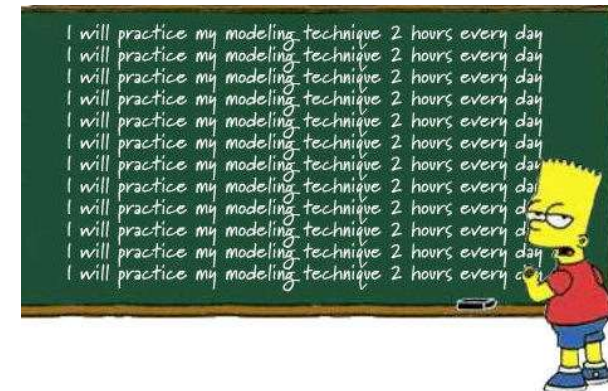
## Using `v-model` on radio buttons

```
<input type="radio" v-model="...">
```



# Workshop v-model

- Create a component with 2 input fields. The values you type in one field, are copied to the other field and vice versa
- Add checkboxes to your own data list. If a field is checked, it is added to an array and shown in the user interface
- **Optional:** create a textfield on one component.
  - Text that is typed in, is passed on as a `prop` to another component
  - See default `HelloWorld` component as an example for `props`
- Examples: [.../125-v-model](#), [126-...](#), [127-...](#), [128-...](#)



# Optional - modifiers for v-model

- Modifying the input, received from a `v-model` textbox
  - `.lazy`
  - `.number`
  - `.trim`
- <https://vuejs.org/v2/guide/forms.html#Modifiers>

## Modifiers

### # `.lazy`

By default, `v-model` syncs the input with the data after each `input` event (with the exception of IME composition as [stated above](#)). You can add the `lazy` modifier to instead sync after `change` events:

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

HTML

### # `.number`

If you want user input to be automatically typecast as a number, you can add the `number`







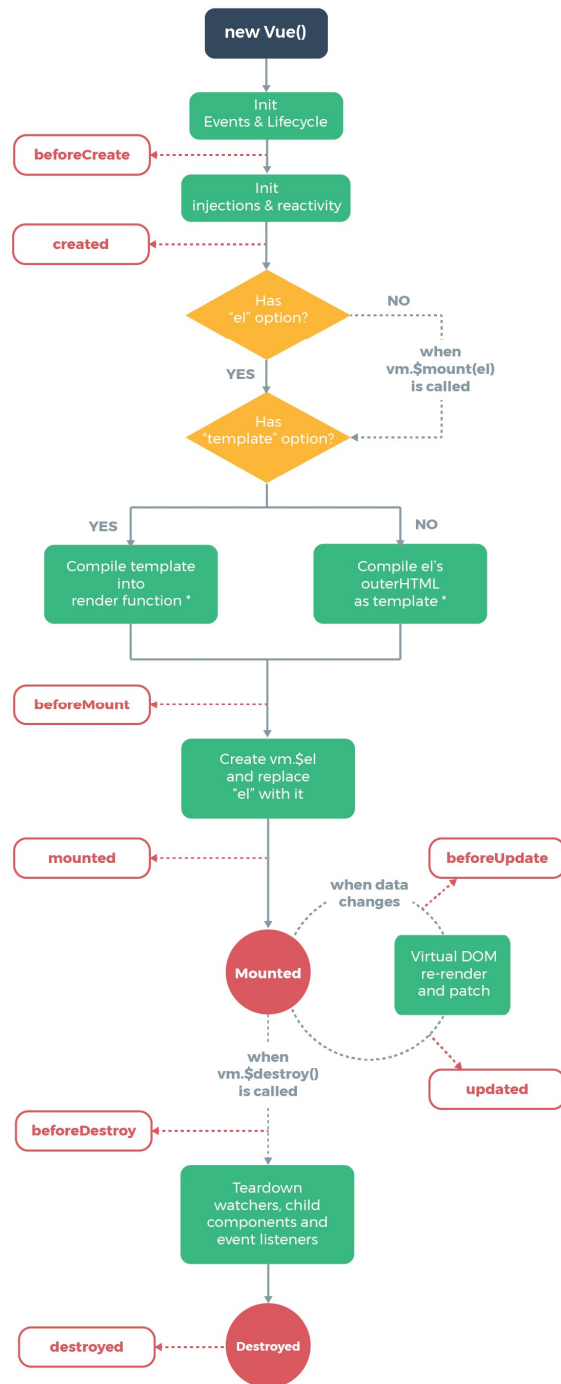
# Component lifecycle hooks

Tapping into the lifecycle of created components

# Lifecycle hooks

- Perform an action automatically when a specific lifecycle event occurs

*“Each component instance goes through a series of initialization steps when it’s created - for example, it needs to set up data observation, compile the template, mount the instance to the DOM, and update the DOM when data changes.”*



## Official lifecycle diagram

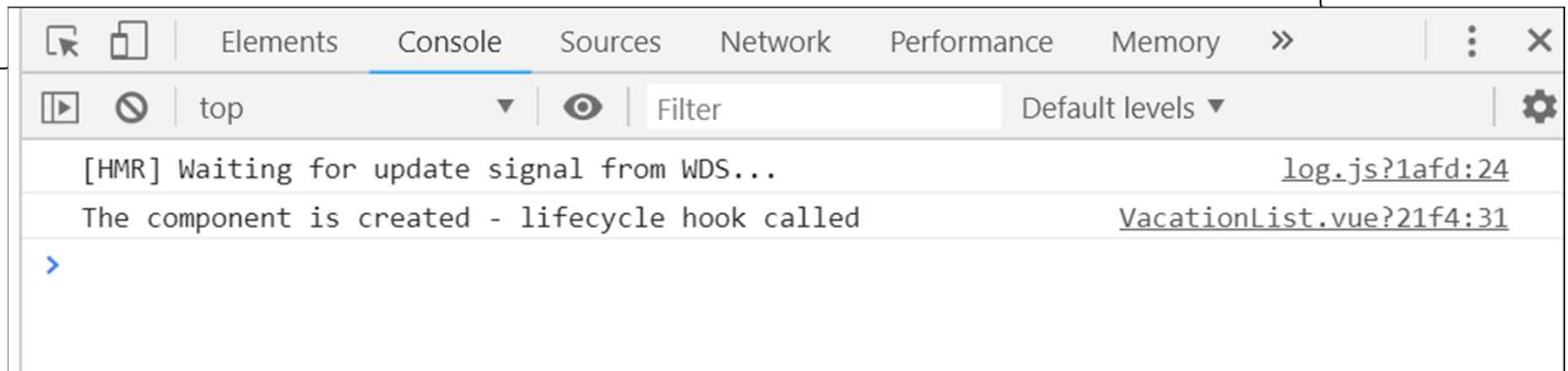
The Red squares are the lifecycle hook methods.

Most used:

- created
- updated
- destroyed

# Using the created hook

```
export default {
  name: "VacationList",
  data() {
    return {
      header: 'List of destinations',
    }
  },
  // Using the 'created' lifecycle hook.
  created(){
    console.log('The component is created - lifecycle hook called');
    // update the header
    this.header = 'The component is created';
  },
  ...
}
```





# Usage of lifecycle hooks

- Typical usage
  - `created` – initialisation of variables, call API's for fetching data etc.
  - `mounted` – if you want to access or modify the DOM.
  - `updated` – when the component receives new data from the outside (props)
  - `destroyed` – to destroy or garbage collect stuff that is not removed automatically
  - **Vue 3:** `unmounted()` (instead of `destroyed()`)

# Workshop

- Create a new component.
- Give it some data that you bind in the UI.
- Use a lifecycle hook `created` to log to the console that the component is created.
- Edit the data in the `created` lifecycle hook. Verify that it is shown correctly in the UI.
- Read the documentation on some other lifecycle hooks, for instance <https://www.digitalocean.com/community/tutorials/vuejs-component-lifecycle>
- Example: [.../150-lifecycle-hooks](#)

