

# VUE ESSENTIALS CHEAT SHEET

## EXPRESSIONS

```
<div id="app">
  <p>I have a {{ product }}</p>
  <p>{{ product + 's' }}</p>
  <p>{{ isWorking ? 'YES' : 'NO' }}</p>
  <p>{{ product.getSalePrice() }}</p>
</div>
```

## DIRECTIVES

Element inserted/removed based on truthiness:

```
<p v-if="inStock">{{ product }}</p>
```

```
<p v-else-if="onSale">...</p>
<p v-else>...</p>
```

Toggles the display: none CSS property:

```
<p v-show="showProductDetails">...</p>
```

Two-way data binding:

```
<input v-model="firstName" >
```

`v-model.lazy="..."` Syncs input after change event

`v-model.number="..."` Always returns a number

`v-model.trim="..."` Strips whitespace

## LIST RENDERING

```
<li v-for="item in items" :key="item.id">
  {{ item }}
</li>
```

*key always recommended*

To access the position in the array:

```
<li v-for="(item, index) in items">...
```

To iterate through objects:

```
<li v-for="(value, key) in object">...
```

Using v-for with a component:

```
<cart-product v-for="item in products"
  :product="item"
  :key="item.id" />
```

## BINDING



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

shorthand

```
<a :href="url">...</a>
```

True or false will add or remove attribute:

```
<button :disabled="isButtonDisabled">...
```

If isActive is truthy, the class 'active' will appear:

```
<div :class="{ active: isActive }">...
```

Style color set to value of activeColor:

```
<div :style="{ color: activeColor }">
```

## ACTIONS / EVENTS

Calls addToCart method on component:



```
<button v-on:click="addToCart">...
```

shorthand

```
<button @click="addToCart">...
```

Arguments can be passed:

```
<button @click="addToCart(product)">...
```

To prevent default behavior (e.g. page reload):

```
<form @submit.prevent="addProduct">...
```

Only trigger once:

```
<img @mouseover.once="showImage">...
```

`.stop` Stop all event propagation

`.self` Only trigger if event.target is element itself

Keyboard entry example:

```
<input @keyup.enter="submit">
```

Call onCopy when control-c is pressed:

```
<input @keyup.ctrl.c="onCopy">
```

Key modifiers:

<code>.tab</code>	<code>.up</code>	<code>.ctrl</code>
<code>.delete</code>	<code>.down</code>	<code>.alt</code>
<code>.esc</code>	<code>.left</code>	<code>.shift</code>
<code>.space</code>	<code>.right</code>	<code>.meta</code>

Mouse modifiers:

<code>.left</code>	<code>.right</code>	<code>.middle</code>
--------------------	---------------------	----------------------



# VUE ESSENTIALS CHEAT SHEET

## COMPONENT ANATOMY



```
Vue.component('my-component', {  
  components: { Components that can be used in the template  
    ProductComponent, ReviewComponent  
  },  
  props: { → The parameters the component accepts  
    message: String,  
    product: Object,  
    email: {  
      type: String,  
      required: true,  
      default: 'none'  
      validator: function (value) {  
        Should return true if value is valid  
      }  
    }  
  },  
  data: function() { Must be a function  
    return {  
      firstName: 'Vue',  
      lastName: 'Mastery'  
    }  
  },  
  computed: { Return cached values until  
    fullName: function () { dependencies change  
      return this.firstName + ' ' + this.lastName  
    }  
  },  
  watch: { Called when firstName changes value  
    firstName: function (value, oldValue) { ... }  
  },  
  methods: { ... },  
  template: '<span>{{ message }}</span>',  
}) Can also use backticks for multi-line
```

## CUSTOM EVENTS

Use props (above) to pass data into child components, custom events to pass data to parent elements.

Set listener on component, within its parent:

```
<button-counter v-on:incrementBy="incWithVal">
```

Inside parent component:

```
methods: {  
  incWithVal: function (toAdd) { ... }  
}
```

Inside button-counter template:

```
this.$emit('incrementBy', 5) Data sent up to parent
```

*Custom event name*

## LIFECYCLE HOOKS



beforeCreate  
created  
beforeMount  
mounted

beforeUpdate  
updated  
beforeDestroy  
destroyed

## USING A SINGLE SLOT

Component template:



```
<div>  
  <h2>I'm a title</h2>  
  <slot>  
    Only displayed if no slot content  
  </slot>  
</div>
```

Use of component with data for slot:

```
<my-component>  
  <p>This will go in the slot</p>  
</my-component>
```

## MULTIPLE SLOTS

Component template:

```
<div class="container">  
  <header>  
    <slot name="header"></slot>  
  </header>  
  <main>  
    <slot>Default content</slot>  
  </main>  
  <footer>  
    <slot name="footer"></slot>  
  </footer>  
</div>
```

Use of component with data for slot:

```
<app-layout>  
  <template v-slot:header><h1>Title</h1></template>  
  <p>The main content.</p>  
  <template v-slot:footer><p>Footer</p></template>  
</app-layout>
```

## LIBRARIES YOU SHOULD KNOW

### Vue CLI

Command line interface for rapid Vue development.

### Vue Router

Navigation for a Single-Page Application.

### Vue DevTools

Browser extension for debugging Vue applications.

### Nuxt.js

Library for server side rendering, code-splitting, hot-reloading, static generation and more.

