# VUE ESSENTIALS CHEAT SHEET



### **EXPRESSIONS**

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

#### **DIRECTIVES**

Element inserted/removed based on truthiness:

```
{{ product }}
```

```
...
...
```

Toggles the display: none CSS property:

```
...
```

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

```
    {{ item }}

    key always recommended
```

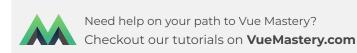
To access the position in the array:

```
...
```

To iterate through objects:

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

Using v-for with a component:



### **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:





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 all event propagation

.self

.stop

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:

.tab	.up	.ctrl	
.delete	. down	.alt	
.esc	.left	.shift	
.space	.right	.meta	

Mouse modifiers:

.left .right .mi	.ddle
------------------	-------

# 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) {
     3 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) { ... }
```

#### **CUSTOM EVENTS**

methods: { ... },

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

template: '<span>{{ message }}</span>',

Can also use backticks for multi-line

Set listener on component, within its parent:

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

Inside parent component:

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

Inside button-counter template:

Custom event name

this. semit ('increment By', 5) Data sent up to parent



Created by your friends at **VueMastery.com** 

## LIFECYCLE HOOKS



beforeCreate beforeUpdate created updated beforeMount beforeDestroy destroyed

## **USING A SINGLE SLOT**

Component template:



Use of component with data for slot:

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

Use of component with data for slot:

```
<app-layout>
<h1 slot="header">Page title</h1>
the main content.
Contact info
</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.