

Table of Contents



- [1. Installation and Setup](#)
 - [Option 1: CDN \(Quick Start\)](#)
 - [Option 2: Vue CLI \(Full Project Setup\)](#)
- [2. Project Structure \(Vue CLI\)](#)
- [3. Basic Component Structure](#)
- [4. Data Binding](#)
 - [1. Interpolation \(Text Binding\):](#)
 - [2. Attribute Binding \(v-bind\):](#)
- [5. Event Handling](#)
- [6. Form Binding \(v-model\)](#)
- [7. Conditional Rendering](#)
- [8. List Rendering](#)
- [9. Components](#)
 - [Parent Component \(App.vue\):](#)
 - [Child Component \(ChildComponent.vue\):](#)
- [10. Computed Properties](#)
- [11. Watchers](#)
- [12. Lifecycle Hooks](#)
- [13. Class and Style Binding](#)
 - [1. Dynamic Classes:](#)
 - [2. Inline Styles:](#)
- [14. Routing \(Vue Router\)](#)
- [15. State Management \(Vuex\)](#)
- [16. Useful CLI Commands](#)

1. Installation and Setup

Option 1: CDN (Quick Start)

```
<script src="https://cdn.jsdelivr.net/npm/vue@3.2.36"></script>
<div id="app">{{ message }}</div>
```

```
<script>
const app = Vue.createApp({
```

```
    data() {
      return {
        message: 'Hello, Vue!'
      }
    }
  }).mount('#app');
</script>
```

Option 2: Vue CLI (Full Project Setup)

1. Install Vue CLI (Globally):

```
npm install -g @vue/cli
```

2. Create New Project:

```
vue create my-vue-app
cd my-vue-app
npm run serve
```

3. Visit:

```
http://localhost:8080
```

2. Project Structure (Vue CLI)

```
my-vue-app/
├── public/           # Static files
├── src/              # Main app code
│   ├── components/  # Vue components
│   ├── App.vue       # Root component
│   └── main.js       # App entry point
├── package.json      # Project dependencies
└── README.md
```

3. Basic Component Structure

App.vue (Single File Component):

```
<template>
  <div>
    <h1>{{ message }}</h1>
  </div>
</template>

<script>
export default {
  data() {
    return {
      message: 'Welcome to Vue.js'
    }
  }
}
</script>

<style>
h1 {
  color: #42b983;
}
</style>
```

4. Data Binding

1. Interpolation (Text Binding):

```
<h1>{{ message }}</h1>
```

2. Attribute Binding (v-bind):

```

```

or shorthand:

```

```

5. Event Handling

```
<button @click="increment">Click Me</button>
<p>Count: {{ count }}</p>
```

```
<script>
export default {
  data() {
    return { count: 0 }
  },
  methods: {
    increment() {
      this.count++;
    }
  }
}
</script>
```

6. Form Binding (v-model)

```
<input v-model="username" placeholder="Enter your name">
<p>Username: {{ username }}</p>
```

```
<script>
export default {
  data() {
```

```
    return { username: '' }  
  }  
}  
</script>
```

7. Conditional Rendering

```
<p v-if="isLoggedIn">Welcome Back!</p>  
<p v-else>Please log in.</p>
```

```
<script>  
export default {  
  data() {  
    return { isLoggedIn: false }  
  }  
}  
</script>
```

8. List Rendering

```
<ul>  
  <li v-for="(item, index) in items" :key="index">  
    {{ item }}  
  </li>  
</ul>
```

```
<script>  
export default {  
  data() {  
    return {  
      items: ['Apple', 'Banana', 'Cherry']  
    }  
  }  
}
```

```
</script>
```

9. Components

Parent Component (App.vue):

```
<template>
  <div>
    <ChildComponent message="Hello from Parent!" />
  </div>
</template>

<script>
import ChildComponent from './components/ChildComponent.vue';
export default {
  components: {
    ChildComponent
  }
}
</script>
```

Child Component (ChildComponent.vue):

```
<template>
  <h2>{{ message }}</h2>
</template>

<script>
export default {
  props: ['message']
}
</script>
```

10. Computed Properties

<p>Reversed Message: {{ reversedMessage }}</p>

```
<script>
export default {
  data() {
    return { message: 'Vue.js' }
  },
  computed: {
    reversedMessage() {
      return this.message.split('').reverse().join('');
    }
  }
}
</script>
```

11. Watchers

<p>Counter: {{ count }}</p>

```
<script>
export default {
  data() {
    return { count: 0 }
  },
  watch: {
    count(newVal, oldVal) {
      console.log(`Count changed from ${oldVal} to ${newVal}`);
    }
  }
}
</script>
```

12. Lifecycle Hooks

```
export default {
  created() {
    console.log('Component Created');
  },
  mounted() {
    console.log('Component Mounted');
  },
  updated() {
    console.log('Component Updated');
  },
  beforeUnmount() {
    console.log('Component Unmounting');
  }
}
```

13. Class and Style Binding

1. Dynamic Classes:

```
<div :class="{ active: isActive }">Dynamic Class</div>

data() {
  return { isActive: true }
}
```

2. Inline Styles:

```
<div :style="{ color: activeColor, fontSize: fontSize + 'px' }">Styled
Text</div>

data() {
  return {
    activeColor: 'red',
    fontSize: 20
  }
}
```

```
}
```

14. Routing (Vue Router)

1. Install Vue Router:

```
npm install vue-router@4
```

2. Setup Routes (router/index.js):

```
import { createRouter, createWebHistory } from 'vue-router';
import Home from '../views/Home.vue';
import About from '../views/About.vue';
```

```
const routes = [
  { path: '/', component: Home },
  { path: '/about', component: About }
];
```

```
export default createRouter({
  history: createWebHistory(),
  routes
});
```

3. Use Router in App:

```
import { createApp } from 'vue';
import App from './App.vue';
import router from './router';
```

```
createApp(App).use(router).mount('#app');
```

15. State Management (Vuex)

1. Install Vuex:

```
npm install vuex@4
```

2. Setup Store (store/index.js):

```
import { createStore } from 'vuex';

export default createStore({
  state() {
    return {
      count: 0
    }
  },
  mutations: {
    increment(state) {
      state.count++;
    }
  }
});
```

3. Use Store in App:

```
import { createApp } from 'vue';
import App from './App.vue';
import store from './store';

createApp(App).use(store).mount('#app');
```

16. Useful CLI Commands

npm run serve	# Start development server
npm run build	# Build for production
npm run lint	# Lint and fix files

This cheat sheet provides the essential building blocks to get started with Vue.js and create dynamic web applications