In Vue, we use the v-if directive to achieve the same:

|  |
| --- |
| <h1 v-if="ok">Yes</h1> |

It is also possible to add an “else block” with v-else:

|  |
| --- |
| <h1 v-if="ok">Yes</h1> <h1 v-else>No</h1> |

F = true

If(f==true)

{

}

Else

{

}

If(f)

{

}

Else

{

}

If(!f)

{

}

Else

{

}

**Conditional Groups with v-if on <template>**

Because v-if is a directive, it has to be attached to a single element. But what if we want to toggle more than one element? In this case we can use v-if on a <template> element, which serves as an invisible wrapper. The final rendered result will not include the <template> element.

|  |
| --- |
| <template v-if="ok">  <h1>Title</h1>  <p>Paragraph 1</p>  <p>Paragraph 2</p> </template> |

**v-else**

You can use the v-else directive to indicate an “else block” for v-if:

|  |
| --- |
| <div v-if="Math.random() > 0.5">  Now you see me </div> <div v-else>  Now you don't </div> |

A v-else element must immediately follow a v-if or a v-else-if element - otherwise it will not be recognized.

**v-else-if**

**New in 2.1.0+**

The v-else-if, as the name suggests, serves as an “else if block” for v-if. It can also be chained multiple times:

|  |
| --- |
| <div v-if="type === 'A'">  A </div> <div v-else-if="type === 'B'">  B </div> <div v-else-if="type === 'C'">  C </div> <div v-else>  Not A/B/C </div> |

Similar to v-else, a v-else-if element must immediately follow a v-if or a v-else-if element.

**Controlling Reusable Elements with key**

Vue tries to render elements as efficiently as possible, often re-using them instead of rendering from scratch. Beyond helping make Vue very fast, this can have some useful advantages. For example, if you allow users to toggle between multiple login types:

|  |
| --- |
| <template v-if="loginType === 'username'">  <label>Username</label>  <input placeholder="Enter your username"> </template> <template v-else>  <label>Email</label>  <input placeholder="Enter your email address"> </template> |

Then switching the loginType in the code above will not erase what the user has already entered. Since both templates use the same elements, the <input> is not replaced - just its placeholder.

Check it out for yourself by entering some text in the input, then pressing the toggle button:

Username



Toggle login type

This isn’t always desirable though, so Vue offers a way for you to say, “These two elements are completely separate - don’t re-use them.” Add a key attribute with unique values:

|  |
| --- |
| <template v-if="loginType === 'username'">  <label>Username</label>  <input placeholder="Enter your username" key="username-input"> </template> <template v-else>  <label>Email</label>  <input placeholder="Enter your email address" key="email-input"> </template> |

Now those inputs will be rendered from scratch each time you toggle. See for yourself:

Username



Toggle login type

Note that the <label> elements are still efficiently re-used, because they don’t have keyattributes.

**v-show**

Another option for conditionally displaying an element is the v-show directive. The usage is largely the same:

|  |
| --- |
| <h1 v-show="ok">Hello!</h1> |

The difference is that an element with v-show will always be rendered and remain in the DOM; v-show only toggles the display CSS property of the element.

Note that v-show doesn’t support the <template> element, nor does it work with v-else.

**v-if vs v-show**

v-if is “real” conditional rendering because it ensures that event listeners and child components inside the conditional block are properly destroyed and re-created during toggles.

v-if is also **lazy**: if the condition is false on initial render, it will not do anything - the conditional block won’t be rendered until the condition becomes true for the first time.

In comparison, v-show is much simpler - the element is always rendered regardless of initial condition, with CSS-based toggling.

Generally speaking, v-if has higher toggle costs while v-show has higher initial render costs. So prefer v-show if you need to toggle something very often, and prefer v-if if the condition is unlikely to change at runtime.