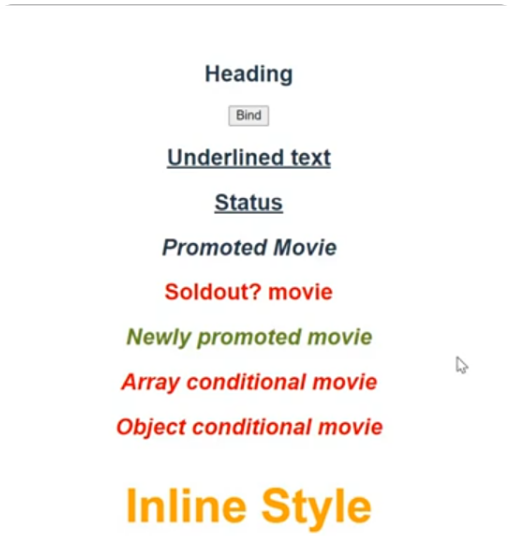
# 28 Jul 2022

## Vue 3 Binding styles

Binding styles

Color is single property

Font size is two words hence written in double codes or else in camel case, styles can also be applied directly static way. Here color, font size are style properties.




```
<h2 v-bind:style="{
  color: highlightColor,
  fontSize: headerSize + 'px',
  padding: '20px'
}">Inline Style</h2>
```

```
<script>
export default {
  name: 'App',
  data() {
    return {
      headingId: 'heading',
      isDisabled: false,
      status: 'success',
      isPromoted: true,
      isSoldout: true,
      highlightColor: 'orange',
      headerSize: 50
    }
  },
}
</script>
```

Object type style binding

```
<h2 v-bind:style="headerStyleObject">Style Object</h2>
```

Array type style binding in array binding last object styling overides first one

```
<h2 v-bind:style="headerStyleObject">Style Object</h2>

<div v-bind:style="[baseStyleObject, successStyleObject]">Success Style</div>
<div v-bind:style="[baseStyleObject, dangerStyleObject]">Danger Style</div>
</template>
```

```
  },
  dangerStyleObject: {
    color: 'darkred',
    backgroundColor: 'red',
    border: '1px solid darkred',
  },
}
```

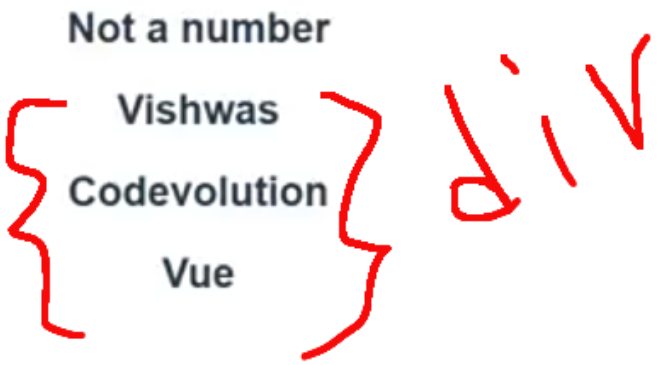
# 28 Jul 2022

## Vue condition rendering 1

V-else always should be followed by v -if /v if-else

If we want to render two parameters then add v-if inside div tag (considered as template not as div)

Instead of div we should bind inside template.




```
<h2 v-else>Not a number</h2>

<div v-if="display">
  <h2>Vishwas</h2>
  <h2>Codevolution</h2>
  <h2>Vue</h2>
</div>
</template>
```

<script>

display: true

</script>

Right way: <template v-if="display">

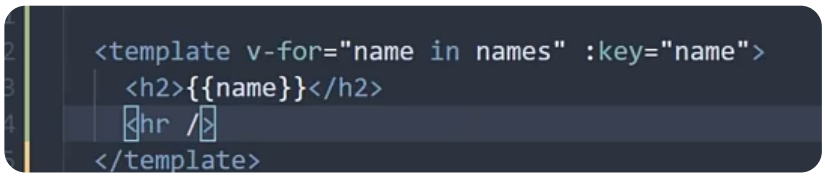
List Rendering 1

Key value and printing value should be same

Keyvalue = iterator value

```
export default {
name: 'App',
data() {
  return {
    names: ['Bruce', 'Clark', 'Diana'],
    fullNames: [
      { first: 'Bruce', last: 'Wayne' },
      { first: 'Clark', last: 'Kent' },
      { first: 'Princess', last: 'Diana' },
    ],
```

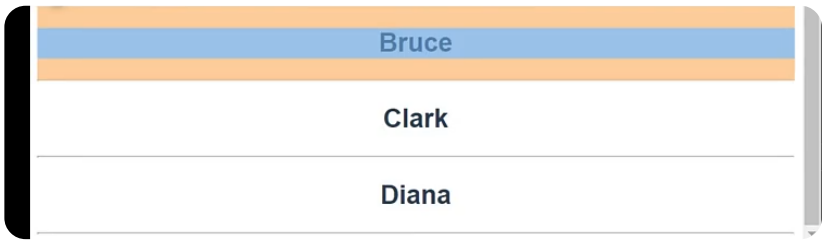
If we want to use a separator between elements then



# Vue condition rendering 2

If we want to use a separator between elements then

The loop works in such away that he is rendered 3 times even we are declaring it only 1 time




```
<template v-for="name in names" :key="name">
  <h2>{{name}}</h2>
  <hr />
</template>
```

```
export default {
  name: 'App',
  data() {
    return {
      names: ['Bruce', 'Clark', 'Diana'],
      fullNames: [
        { first: 'Bruce', last: 'Wayne' },
        { first: 'Clark', last: 'Kent' },
        { first: 'Princess', last: 'Diana' },
      ],
      actors: [
        {
          name: 'Christian Bale',
          movies: ['Batman', 'The Prestige'],
        },
        {
          name: 'Di Caprio',
          movies: ['Titanic', 'Inception'],
        },
      ],
      myInfo: {
        name: 'Vishwas',
        channel: 'Codevolution',
        course: 'Vue 3'
      }
```

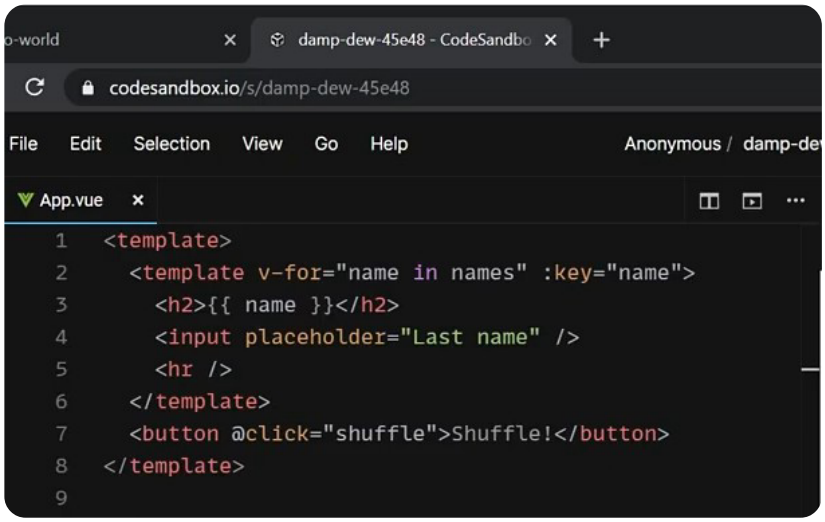
If we want to display particular element in a loop

Then the ideal way is as follows only bruce is displayed

```vue
<template>
  <template v-for="name in names" :key="name">
    <h2 v-if="name === 'Bruce'">{{ name }}</h2>
  </template>
</template>

<script>
export default {
  name: 'App',
  data() {
    return {
      names: ['Bruce', 'Clark', 'Diana'],
    }
  },
}
</script>
```

Use of key attribute in Vue

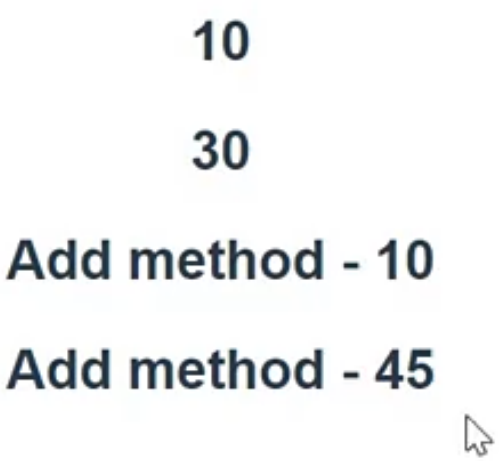



If we use key then ok only even after shuffling particular input sticked with particular element

# Vue event handling

```
<template>
  <h2>{{ 2 + 3 + 5 }}</h2>
  <h2>{{ 5 + 10 + 15 }}</h2>
  <h2>Add method - {{ add() }}</h2>
</template>

<script>
export default {
  name: 'App',
  data() {
    return {}
  },
  methods: {
    add() {
      return 2 + 3 + 5
    },
  },
}
</script>
```




```
<template>
  <h2>{{ 2 + 3 + 5 }}</h2>
  <h2>{{ 5 + 10 + 15 }}</h2>
  <h2>Add method - {{ add(2,3,5) }}</h2>
  <h2>Add method - {{ add(10,15,20) }}</h2>
</template>

<script>
export default {
  name: 'App',
  data() {
    return {}
  },
  methods: {
    add(a, b, c) {
      return a + b + c
    },
  },
}
</script>
```

```
<template>
  <h2>{{name}}</h2>
  <div>
    <button v-on:click="name = 'Batman'">Change name</button>
  </div>
</template>

<script>
export default {
  name: 'App',
  data() {
    return {
      name: 'Vishwas'
    }
  },
  methods: {
  },
}
</script>
```


On click name changes from vishwas to batman

# Event handling 2


```
<div>
  <button @click="increment(1, $event)">Increment 1</button>
```

```
methods: {
  changeName(event) {
    this.name = 'Batman'
    console.log('Event', event)
  },
  increment(num, event) {
    this.count += num
    console.log('Event', event)
  },
```

```
<template>
  <h2>{{name}}</h2>
  <div>
    <button v-on:click="changeName">Change name</button>
  </div>
```

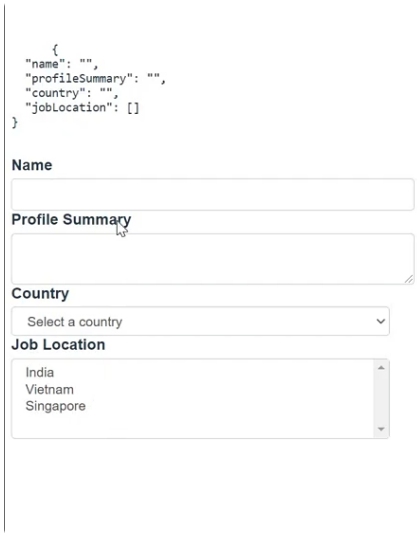
## Form handling 1

Created formvalues and  and gave name,defined that in template and binded with V-Model formvalues.name



To send the data from template to script (2 way binding) we use V-Mlodel

```vue
<script>
export default {
  name: 'App',
  data() {
    return {
      formValues: {
        name: '',
        profileSummary: '',
        country: '',
        jobLocation: []
      },
    }
  },
  methods: {},
}
</script>
```




```vue
    <div>
      <label for="name">Name</label>
      <input type="text" id="name" v-model="formValues.name" />
    </div>

    <div>
      <label for="profile">Profile Summary</label>
      <textarea id="profile" v-model="formValues.profileSummary" />
    </div>

    <div>
      <label for="country">Country</label>
      <select id="country" v-model="formValues.country">
        <option value="">Select a country</option>
        <option value="india">India</option>
        <option value="vietnam">Vietnam</option>
        <option value="singapore">Singapore</option>
      </select>
    </div>

  </form>
</template>
```

```vue
    <div>
      <label for="job-location">Job Location</label>
      <select id="job-location" multiple v-model="formValues.jobLocation">
        <option value="">Select a country</option>
        <option value="india">India</option>
        <option value="vietnam">Vietnam</option>
        <option value="singapore">Singapore</option>
      </select>
    </div>
```

# 30 Jul 2022

## Form handling 2

Skill set html in template

```
<div>
  <label>Skill set</label>
  <input type="checkbox" id="html" value="html" v-model="formValues.skillSet" />
  <label for="html">HTML</label>
  <input type="checkbox" id="css" value="css" v-model="formValues.skillSet" />
  <label for="css">CSS</label>
  <input type="checkbox" id="javascript" value="javascript" v-model="formValues.skillSet" />
  <label for="javascript">JavaScript</label>
</div>
```

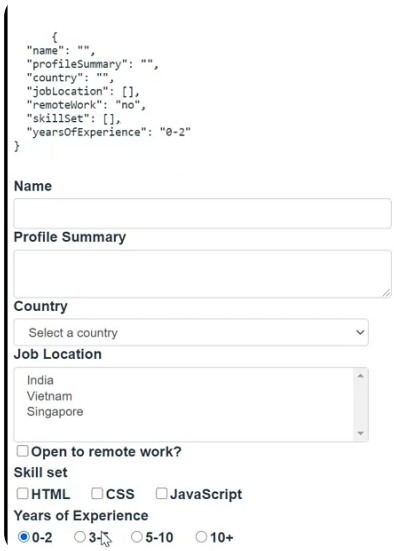
Submit form event passing , by default page gets reloaded to avoid this we give prevent default.

```
    </pre>
  </div>
  <form @submit="submitForm">
    <div>
```

```
        yearsOfExperience: '',
      },
    }
  },
  methods: {
    submitForm(event) {
      event.preventDefault()
      console.log('Form values', this.formValues)
    }
  },
}
</script>
```

This is for radio button and binding using vmodel

```
<div>
  <label>Years of Experience</label>
  <input
    type="radio"
    id="0-2"
    value="0-2"
    v-model="formValues.yearsOfExperience"
  />
  <label for="0-2">0-2</label>
  <input
    type="radio"
    id="3-5"
    value="3-5"
    v-model="formValues.yearsOfExperience"
  />
  <label for="3-5">3-5</label>
  <input
    type="radio"
    id="6-10"
    value="6-10"
    v-model="formValues.yearsOfExperience"
  />
  <label for="6-10">5-10</label>
```




form values attribute inside script

```
<script>
export default {
  name: 'App',
  data() {
    return {
      formValues: {
        name: '',
        profileSummary: '',
        country: '',
        jobLocation: []
      },
    }
  },
  methods: {},
}
</script>
```

## Computed properties

On click of button keyboard price also added to computed total (total is computed property mentioned) and it returns  curr.price has each price of item and goes on adds up and calls.

```
      },
    ],
  }
},
methods: {},
computed: {
  fullName() {
    return `${this.firstName} ${this.lastName}`
  },
  total() {
    return this.items.reduce((total, curr) => (total = total + curr.price), 0)
  }
},
}
</script>
```

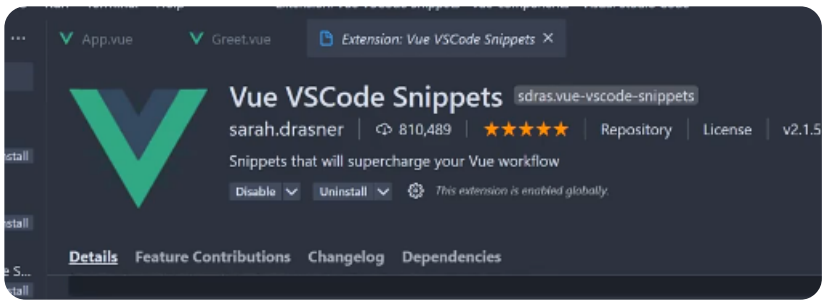

```
<template>
  <h2>Fullname - {{ firstName }} {{ lastName }}</h2>
  <h2>Computed fullname - {{ fullName }}</h2>
  <button @click="items.push({id: 4, title: 'Keyboard', price: 50})">Add item</button>
  <h2>Computed Total - {{ total }}</h2>
</template>

<script>
export default {
  name: 'App',
  data() {
    return {
      firstName: 'Bruce',
      lastName: 'Wayne',
      items: [
        {
          id: 1,
          title: 'TV',
          price: 100,
        },
        {
          id: 2,
          title: 'Phone',
          price: 200,
```

# 01 Aug 2022

## Props

Greet.vue

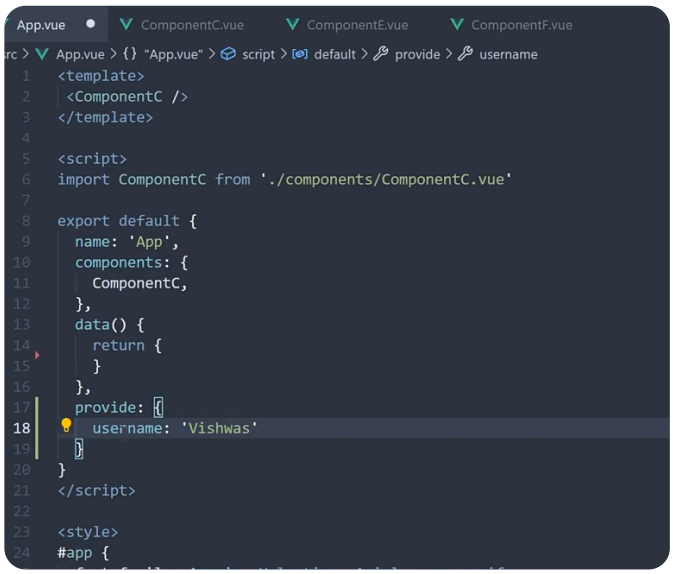



# 01 Aug 2022

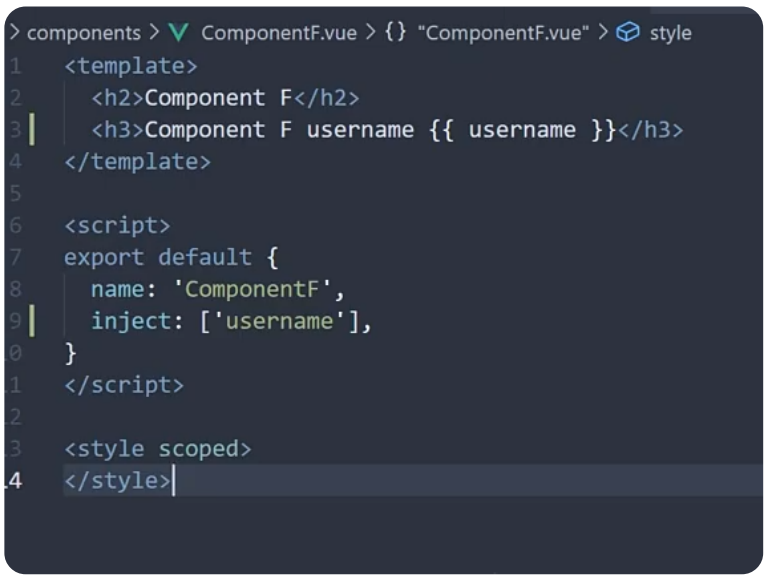
## Provide and inject

Case 1

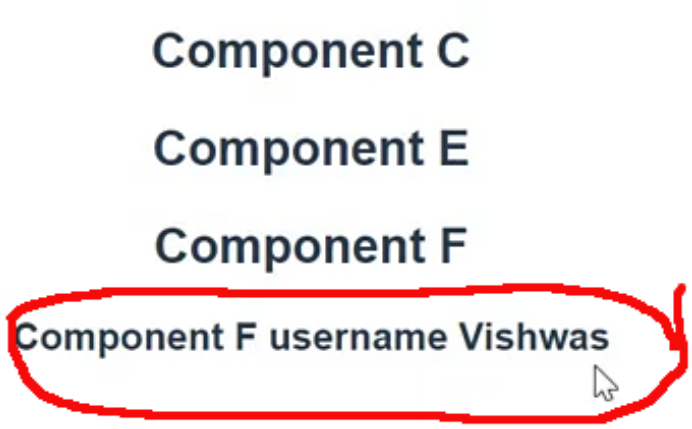
Step 1




Step 2

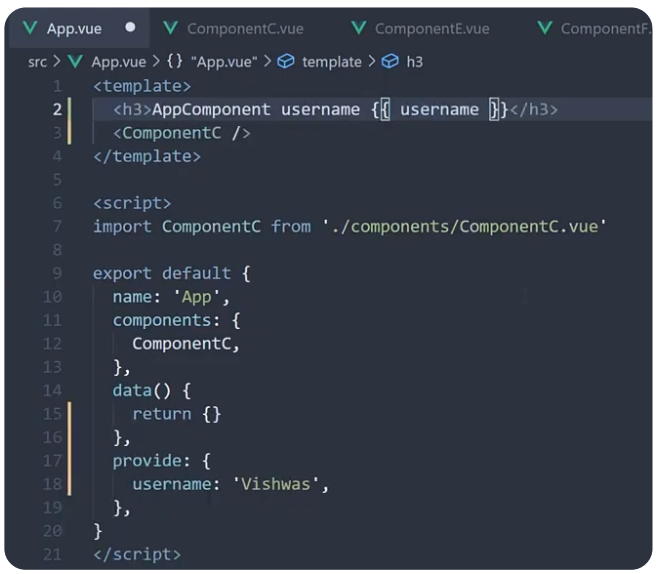



Output




Case 2

If we want to display username into template but username is not displaying in this way



AppComponent username

Component C

Component E

Component F

Component F username Vishwas