What is Vue.js?

Vue is a **progressive framework** for building user interfaces. Unlike other monolithic frameworks, Vue is designed from the ground up to be incrementally adoptable. The core library is focused on the view layer only, and is easy to pick up and integrate with other libraries or existing projects. On the other hand, Vue is also perfectly capable of powering sophisticated Single-Page Applications when used in combination with **modern tooling** and **supporting libraries**.

---

source: https://vuejs.org/v2/guide/

VUE IS

THE BEST

How to use it?

1. **CDN**

```
<script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
```

2. **CLI**

```
vue create my-project
cd my-project
vue add mdb
```

file structure

```
<template>HTML code here</template>

<script>Javascript</script>

<style scoped>Scoped CSS</style>
```

source: https://vuejs.org/v2/guide/

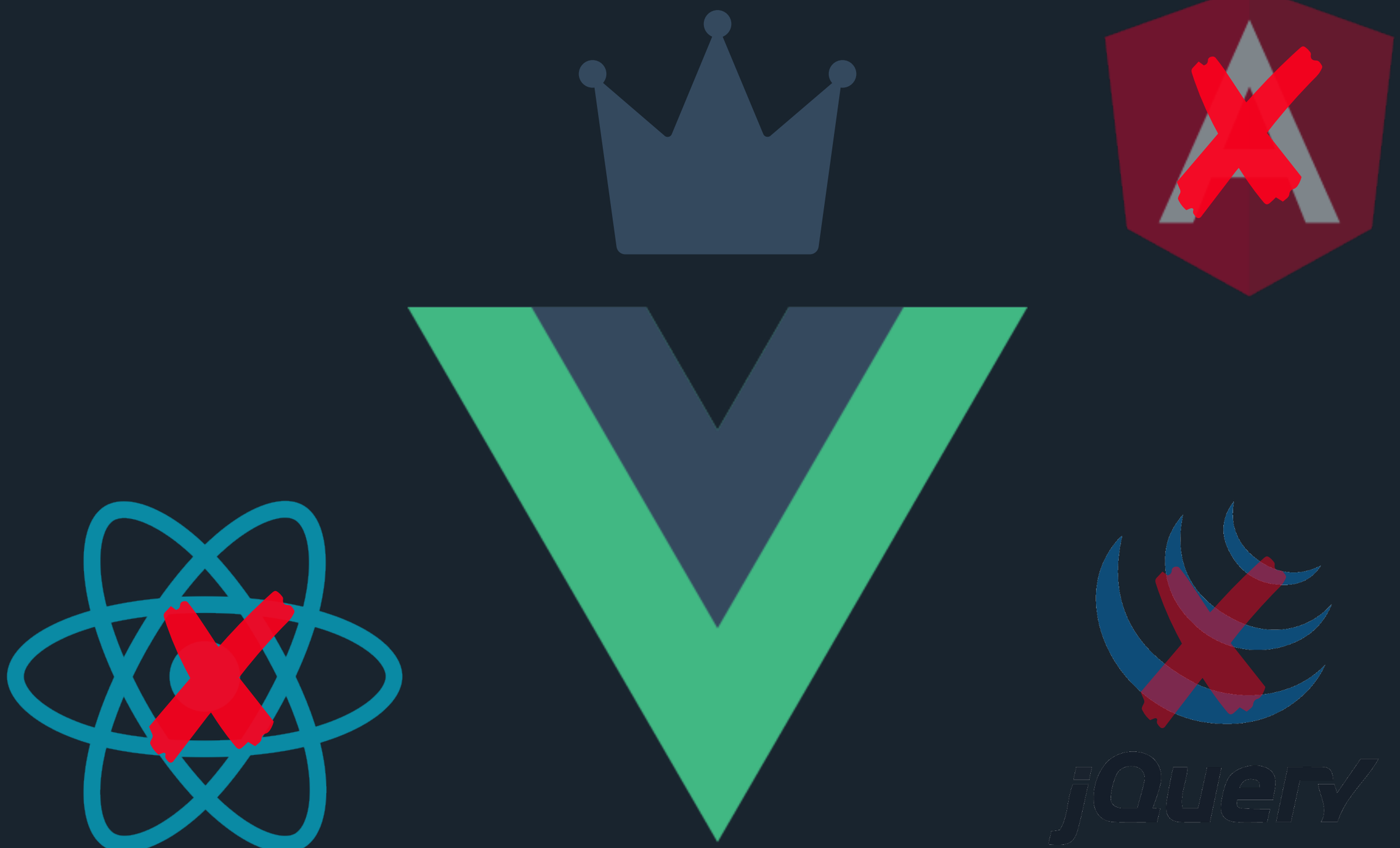
file structure

```
<template><mdb-btn>Click me</mdb-btn></template>

<script>
  import { mdbBtn } from ‘mdbvue’;

  export default {
    name: ‘HelloWorld’,
    components: {
      mdbBtn
    }
  }
</script>

 <style scoped>Scoped CSS</style>
```

---

source: https://vuejs.org/v2/guide/

```
<script>
    import { mdbBtn } from 'mdbvue';
    export default {
        name: 'HelloWorld',
        components: {
            mdbBtn
        },
        data() {
            return {
                name: 'Stefan'
            }
        },
        computed: {
            computedValue() {
                return `hi, ${this.name}`;
            }
         },
        methods: {
            clickHandler() {
                console.log("hi!")
            }
        }
    }
</script>
```


jQuery

lifecycle

beforeCreate()
created()

beforeMount()
mounted()

beforeUpdate()
updated()

beforeDestroy()
destroyed()

v-on / @

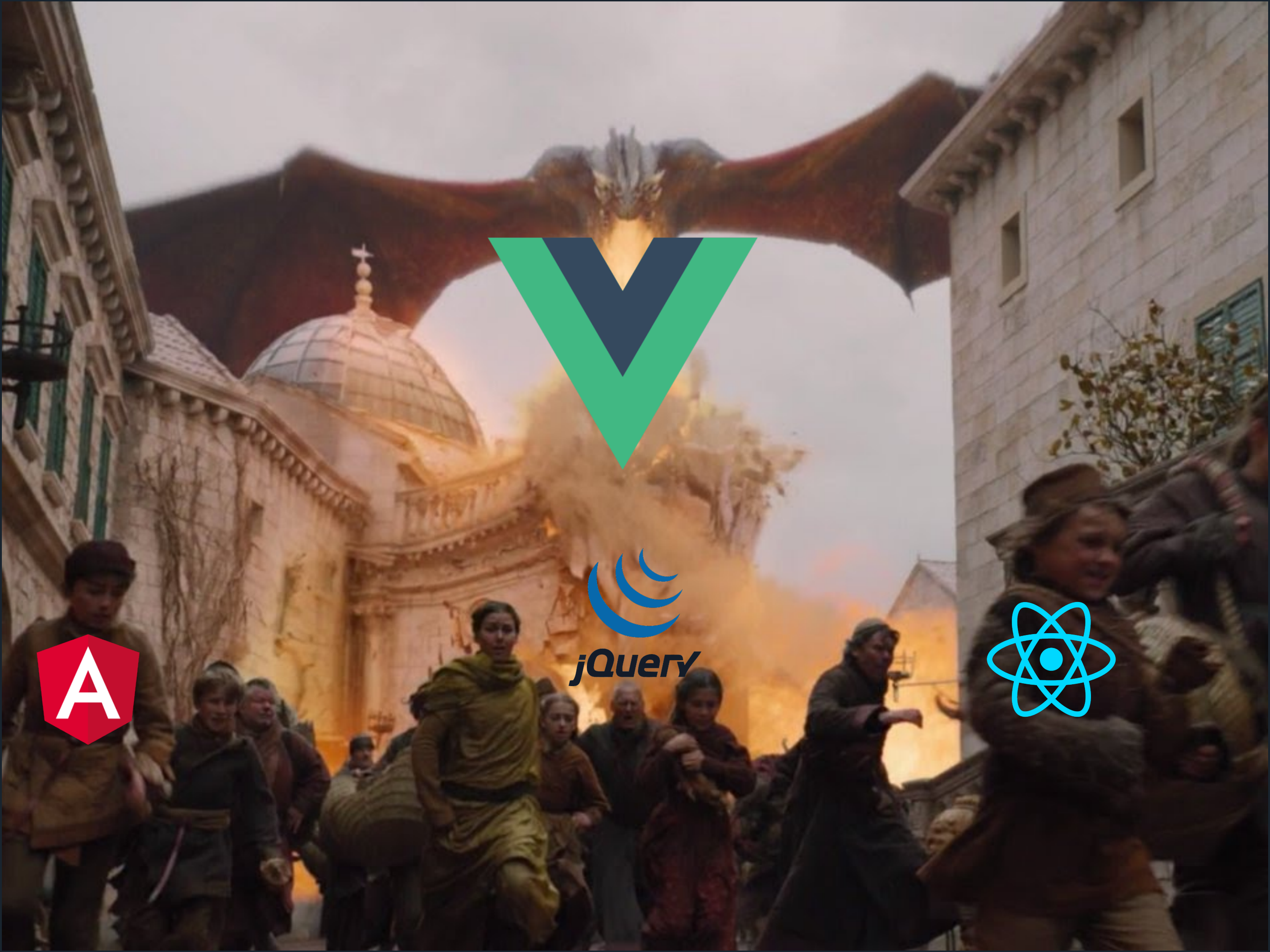
event handling

```
<template>
    <mdb-btn @click="handleClick">Click me</mdb-btn>
</template>

<script>
    import { mdbBtn } from 'mdbvue';
    export default {
        name: 'HelloWorld',
        components: {
            mdbBtn
        },
        methods:  {
            handleClick(e) {
                console.log(e.target);
            }
        }
    }
</script>
```


jQuery

conditional rendering

```
<template>
    <div>
        <h2> Conditional rendering </h2>
        <mdbBtn @click="toggle = !toggle">Toggle text</mdbBtn>
        <p v-if="toggle">First text</p>
        <p v-else>Second text</p>
    </div>
</template>

<script>
import { mdbBtn } from 'mdbvue';

export default {
    name: 'Conditional',
    components: { mdbBtn },
    data() {
        return {
            toggle: false
        };
    }
}
</script>
```


jQuery

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.

source: https://vuejs.org/v2/guide/

looping

```
<template>
    <div>
        <p v-for="(task, i) in tasks" :key="i">{{task}}</p>
    </div>
</template>

<script>
import { mdbBtn } from 'mdbvue';

export default {
    components: { mdbBtn },
    data() {
        return {
            tasks: ['eat', 'sleep', 'code']
        };
    }
}
</script>
```

:class / class

classes

```
<template>
   <p class="zolkusowy-gradient" :class="color === red ? 'red' : 'green'">
        Some pretty text with gradient background
   </p>
</template>

<script>
   export default {
       name: 'HelloWorld',
       props: {
           color: String
       },
       computed: {
         classes() {
               if (this.color === 'red') {
                    return 'red-text danger'
               }
          }
       }
   }
</script>
```

dziękuję za uwagę
i zapraszam do kodzenia