Class 4: Interactive Components

1. Review: Vue.js Events & Reactive Variables
2. Components
3. Props
4. Conditional Rendering
5. Rendering Lists

Review: Event Handling – Vue.js

The `v-on` directive listens for DOM events and runs JavaScript code when they're triggered.

`v-on` Directive

```
<script setup>
function buttonClicked() {
  console.log('Button clicked!')
}
</script>

<template>
  <button v-on:click="buttonClicked">Click Me!</button>
</template>
```

`@` Shorthand

```
<script setup>
function buttonClicked() {
  console.log('Button clicked!')
}
</script>

<template>
  <button @click="buttonClicked">Click Me!</button>
</template>
```

Review: Reactive Variable + Interpolation

```
<script setup>
import { ref } from 'vue'
const count = ref(0)

function incrementCount() {
  count.value++
}
</script>

<template>
  <button @click="incrementCount">
    Increment
  </button>
  <span>{{ count }}</span>
</template>
```

Review: `.vue` Files

- Each `.vue` file is a **component**.
- Each component has:
  - `<script>` : JavaScript logic
  - `<template>` : HTML structure
  - `<style>` : CSS styling

```
<script setup>
</script>

<template>
  <button>Click Me!</button>
</template>

<style scoped>
button {
  font-size: 1.5em;
  padding: 0.5em 1em;
}
</style>
```

Components

User Interface Components

Definition: A **component** is a reusable, self-contained piece of user interface (UI) that can be used to build larger applications.

Examples:

- Button
- Slider
- Search Input
- Dropdown Menu

Review: React Components

`components/StarRating.jsx`:

```
export default function StarRating() {
  return (
    <div>
      <button>'☆'</button>
      <button>'☆'</button>
      <button>'☆'</button>
      <button>'☆'</button>
      <button>'☆'</button>
    </div>
  )
}
```

`App.jsx`:

```
import StarRating from './components/StarRating.jsx'

export default function App() {
  return (
    <>
      <StarRating />
    </>
  )
}
```

Vue.js Components

`components/StarRating.vue`:

```
<script setup>
</script>

<template>
  <div>
    <button>'☆'</button>
    <button>'☆'</button>
    <button>'☆'</button>
    <button>'☆'</button>
    <button>'☆'</button>
  </div>
</template>

<style scoped>
</style>
```

`App.vue`:

```
<script setup>
import StarRating from './components/StarRating.vue'
</script>

<template>
  <StarRating />
</template>
```

Activity: Component Practice

1. Open the activity repository for today's class.
2. Create a new `SoundButton` component with **hard-coded data**. Hardcode the sound name, icon, and sound file into the component. For example: "Rain", `rain.svg`, and `rain.mp3`.
3. The `SoundButton` component should include internally include the ability to play the sound when clicked.
4. Use the `SoundButton` component in your app to replace the existing button for that sound.

Props

Review: React Props

`components/StarRating.jsx`:

```
export default function StarRating({name}) {
  return (
    <div>
      <span>{name}</span>
      <button>'☆'</button>
      <button>'☆'</button>
      <button>'☆'</button>
      <button>'☆'</button>
      <button>'☆'</button>
    </div>
  )
}
```

`App.jsx`:

```
import StarRating from './components/StarRating.jsx'

export default function App() {
  return (
    <>
      <StarRating name="Movie Rating"/>
    </>
  )
}
```

Vue.js Props

`components/StarRating.vue`:

```
<script setup>
defineProps({
  name: String
})
</script>

<template>
  <div>
    <span>{{ name }}</span>
    <button>'☆'</button>
    <button>'☆'</button>
...
```

`App.vue`:

```
<script setup>
import StarRating from './components/StarRating.vue'
</script>

<template>
  <StarRating name="Movie Rating"/>
</template>
```

Activity: Props Practice

1. Modify the `SoundButton` component to accept **props** for the sound name, icon, and sound file.
2. Use the `SoundButton` component in your app to replace the existing button for that sound, passing the appropriate props to the component.

```
<script setup>
defineProps({
  name: String
})
</script>
```

Gotcha: Binding Props

Literal values:

```
<StarRating name="Movie Rating"/>
```

Variables, Reactive Variables, Expressions:

```
<script setup>
import { ref } from 'vue'
const movieRating = ref('Movie Rating')
</script>

<template>
  <StarRating :name="movieRating"/>
</template>
```

```
<script setup>
import { ref } from 'vue'
const movieRating = ref('Movie Rating')
</script>

<template>
  <StarRating v-bind:name="movieRating"/>
</template>
```

Comparison: React vs Vue.js

React

```
export default function App() {
  const movieRating = "Movie Rating"

  return (
    <>
      {/* string literal */}
      <StarRating name="Movie Rating"/>

      {/* variable */}
      <StarRating name={movieRating}/>
    </>
  )
}
```

Vue.js

```
<script setup>
import { ref } from 'vue'
const movieRating = ref('Movie Rating')
</script>

<template>
  <!-- string literal -->
  <StarRating name="Movie Rating"/>

  <!-- variable -->
  <StarRating :name="movieRating"/>
</template>
```

Conditional Rendering

Review: React Conditional Rendering

`components/StarRating.jsx`:

```
export default function StarRating({name}) {
  return (
    <div>
      {!!name && <span>{name}</span>}
      <button>'☆'</button>
      <button>'☆'</button>
      <button>'☆'</button>
      <button>'☆'</button>
      <button>'☆'</button>
    </div>
  )
}
```

Vue.js Conditional Rendering

`components/StarRating.vue`:

```
<script setup>
defineProps({
  name: String
})
</script>

<template>
  <div>
    <span v-if="!!name">{{ name }}</span>
    <button>'☆'</button>
...
```

Activity: Conditional Rendering Practice

Working with your **peer**, complete the **handout**.

```
<script setup>
defineProps({
  name: String
})
</script>

<template>
  <div>
    <span v-if="!!name">{{ name }}</span>
    <button>'☆'</button>
...
```

Rendering Lists

Review: React Lists

`components/StarRating.jsx`:

```
export default function StarRating({name}) {
  return (
    <div>
      <span>{name}</span>
      {[1, 2, 3, 4, 5].map((num) => (
        <button key={num}>'☆'</button>
      ))}
    </div>
  )
}
```

Vue.js `v-for` Directive

`components/StarRating.vue`:

```
<script setup>
defineProps({
  name: String
})
</script>

<template>
  <div>
    <span>{{ name }}</span>
    <button v-for="num in 5" :key="num">'☆'</button>
  </div>
</template>
```

Activity: Rendering Lists Practice

Render a list of `SoundButton` components in your app using the `v-for` directive. (They should all be the **same** sound button for this activity.)

```
<template>
  <button v-for="num in 5" :key="num">'☆'</button>
</template>
```

What's Next

Render different `SoundButton` components in your app using the `v-for` and reactive variables.

- Due Monday, 9/7/2026, 11:59PM: Class 5, Preparation