Build a Modern Single Page Application with Vue

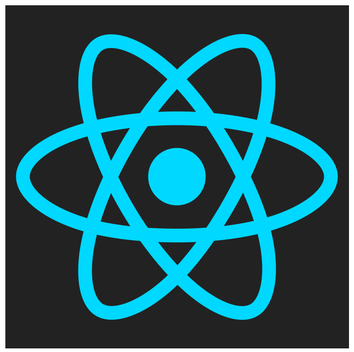
https://www.mattburkedev.com/vue-workshop/

Matt Burke • Software Architect @ ABOUT Healthcare

- Install and Setup
- Templating
- Forms & Reactivity
- Components
- Routing & Navigation
- Data & State Management

https://www.mattburkedev.com/vue-workshop/

A word on ~~copy/paste~~CoPilot



Component-Based Frameworks


About
Store
Gmail
Images
Google
Google Search
I'm Feeling Lucky
Advertising
Business
How Search works
Our third decade of climate action: join us
Privacy
Terms
Settings


About
Store
Gmail
Images
Google
Google Search
I'm Feeling Lucky
Advertising
Business
How Search works
Our third decade of climate action: join us
Privacy
Terms
Settings


Google
Google Search
I'm Feeling Lucky

Option vs Composition API

Options API

- Older syntax
- Organize by technical concern
- Access properties via `this`
- Poor typescript support

```vue
<script>
export default {
  // Properties returned from data() become reactive state
  // and will be exposed on `this`.
  data() {
    return {
      count: 0
    }
  },

  // Methods are functions that mutate state and trigger updates.
  // They can be bound as event handlers in templates.
  methods: {
    increment() {
      this.count++
    }
  },

  // Lifecycle hooks are called at different stages
  // of a component's lifecycle.
  // This function will be called when the component is mounted.
  mounted() {
    console.log(`The initial count is ${this.count}.`)
  }
}
</script>

<template>
  <button @click="increment">Count is: {{ count }}</button>
</template>
```

Composition API

Recommended Syntax

- Organize by feature
- Support shared logic through composables
- Excellent typescript support
- Better minification

```vue
<script setup>
import { ref, onMounted } from 'vue'

// reactive state
const count = ref(0)

// functions that mutate state and trigger updates
function increment() {
  count.value++
}

// lifecycle hooks
onMounted(() => {
  console.log(`The initial count is ${count.value}.`)
})
</script>

<template>
  <button @click="increment">Count is: {{ count }}</button>
</template>
```

Tooling


npm
esbuild
An extremely fast JavaScript bundler
node
JS

esbuild

Development time bundler

Rollup.js

Production build bundler

```
npm create vite@latest
```

```
<script setup lang=“ts”>

</script>

<template>



</template>

<style scoped>



</style>
```

```
<script setup lang=“ts”>

</script>

<template>



</template>

<style scoped>



</style>
```

```
<script setup lang=“ts”>
const message = “Hello World”
</script>

<template>




</template>

<style scoped>




</style>
```

```
<script setup lang="ts">
const message = "Hello World"
</script>

<template>




</template>

<style scoped>




</style>
```

```
<script setup lang="ts">
const message: string = "Hello World"
</script>

<template>




</template>

<style scoped>




</style>
```

```
<script setup lang="ts">
const message: string = "Hello World"
</script>

<template>



</template>

<style scoped>



</style>
```

```
<script setup lang="ts">
const message: string = "Hello World"
</script>

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

<style scoped>




</style>
```

```
<script setup lang="ts">
const message: string = "Hello World"
</script>

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

<style scoped>
h1 {
  font-weight: bold;
  font-size: 24px
}
</style>
```

```
<script setup lang=“ts”>
const message: string = “Hello World”
</script>

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

<style scoped>
h1 {
  font-weight: bold;
  font-size: 24px
}
</style>
```

https://www.mattburkedev.com/vue-workshop/intro-and-setup/

Templating

```
<script setup>

</script>

<template>

</template>
```

```
<script setup>
const message = “Hello”;
</script>

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

```
<script setup>
const message = “Hello”;
</script>

<template>
  <p>{{ message }}, CodeMash</p>
</template>
```

```
<script setup>
const message = “Hello”;
</script>

<template>
  <p>{{ message }}, CodeMash</p>
</template>
```

```
<script setup>
const message = “Hello”;
</script>


<template>
  <p>{{ message }}, CodeMash</p>
</template>
```

Expressions

```
<template>
  <p>
    {{ message }} CodeMash
  </p>
</template>
```

```
<template>
  <p>
    {{ message.toUpperCase() }} CodeMash
  </p>
</template>
```

Attribute Binding

```
<script setup>
const profile = getProfile();
</script>

<template>
  <img v-bind:src="profile.photoUrl">
</template>
```

```
<script setup>
const profile = getProfile();
</script>

<template>
  <img :src="profile.photoUrl">
</template>
```

```
<script setup>
const profile = getProfile();
</script>

<template>
  <img :src="profile.photoUrl">
</template>
```

```
<script setup>
const profile = getProfile();
</script>

<template>
  <img src=“profile.photoUrl”>
</template>
```

```
<script setup>
const profile = getProfile();
</script>

<template>
  <img src="profile.photoUrl">
</template>
```

```
<script setup>
const profile = getProfile();
</script>

<template>
  <img src="prof                Url">
</template>
```

Style Binding

```
<script setup>
const color = “#FF0000”;
</script>

<template>
<div style=“background-color: #FF0000;”>
  …
</div>
</template>
```

```
<script setup>
const color = “#FF0000”;
</script>

<template>
<div :style=“{ backgroundColor: color }”>
  …
</div>
</template>
```

```
<script setup>
const color = “#FF0000”;
</script>

<template>
<div :style=“{ backgroundColor: color }”>
  …
</div>
</template>
```

Class Binding

```
<script setup>
const isActive = Math.random() > 0.5;
</script>

<template>
<button class="btn btn-active">
  …
</button>
</template>
```

```
<script setup>
const isActive = Math.random() > 0.5;
</script>

<template>
<button class="btn btn-active">
  …
</button>
</template>
```

```
<script setup>
const isActive = Math.random() > 0.5;
</script>

<template>
<button :class="['btn', isActive ? 'btn-active' : '']">
  …
</button>
</template>
```

```
<script setup>
const isActive = Math.random() > 0.5;
</script>

<template>
<button :class="['btn', isActive ? 'btn-active' : '']">
  …
</button>
</template>
```

```
<script setup>
const isActive = Math.random() > 0.5;
</script>

<template>
<button :class="['btn', isActive ? 'btn-active' : '']">
  …
</button>
</template>
```

```
<script setup>
const isActive = Math.random() > 0.5;
</script>

<template>
<button :class="['btn', isActive ? 'btn-active' : '']">
  …
</button>
</template>
```

```
<script setup>
const isActive = Math.random() > 0.5;
</script>

<template>
<button :class="['btn', isActive ? 'btn-active' : '']">
  …
</button>
</template>
```

```
<script setup>
const isActive = Math.random() > 0.5;
</script>

<template>
<button :class="['btn', isActive ? 'btn-active' : '']">
  …
</button>
</template>
```

```
<script setup>
const isActive = Math.random() > 0.5;
</script>

<template>
<button :class="{ btn: true, 'btn-active': isActive }">
  …
</button>
</template>
```

```
<script setup>
const isActive = Math.random() > 0.5;
</script>

<template>
<button :class="{ btn: true, 'btn-active': isActive }">
  …
</button>
</template>
```

```
<script setup>
const isActive = Math.random() > 0.5;
</script>

<template>
<button :class="{ btn: true, 'btn-active': isActive }">
  …
</button>
</template>
```

```
<script setup>
const isActive = Math.random() > 0.5;
</script>

<template>
<button :class="{ btn: true, 'btn-active': isActive }">
  …
</button>
</template>
```

```
<script setup>
const isActive = Math.random() > 0.5;
</script>

<template>
<button class="btn" :class="{'btn-active': isActive }">
  …
</button>
</template>
```

Conditionals

```
<template>
const { loading, data } = makeApiCall();
</template>


<template>
<div v-if="loading">
  <img src="spinner.gif">
</div>
<div v-else>
  …
</div>
</template>
```

```
<template>
const { loading, data } = makeApiCall();
</template>


<template>
<div v-if="loading">
  <img src="spinner.gif">
</div>
<div v-else>
  …
</div>
</template>
```

```
<template>
const { loading, data } = makeApiCall();
</template>


<template>
<div v-if="loading">
  <img src="spinner.gif">
</div>
<div v-else>
  …
</div>
</template>
```

```
<template>
const { loading, data } = makeApiCall();
</template>


<template>
<div v-if="loading">
  <img src="spinner.gif">
</div>
<div v-else>
  ...
</div>
</template>
```

Lists

```
<script setup>

</script>

<template>









</template>
```

```
<script setup>
const friends = […];
</script>

<template>




</template>
```

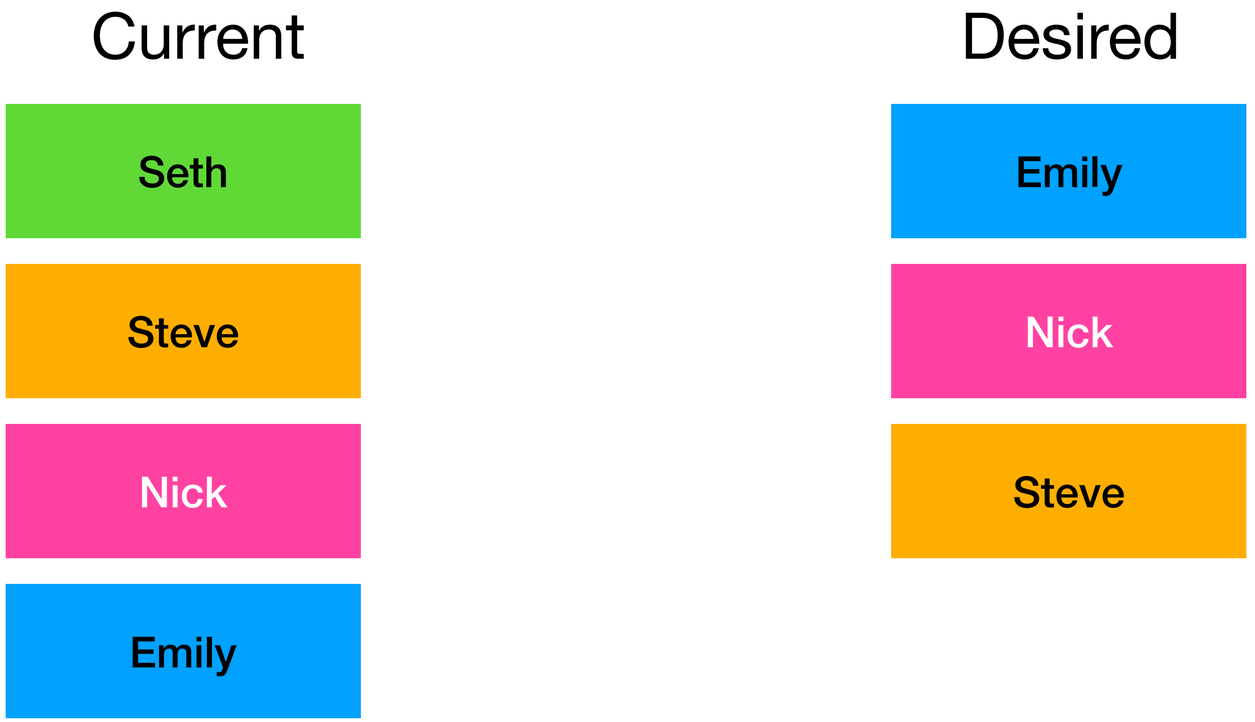
```
<script setup>
const friends = […];
</script>

<template>
<ul>




</ul>

</template>
```

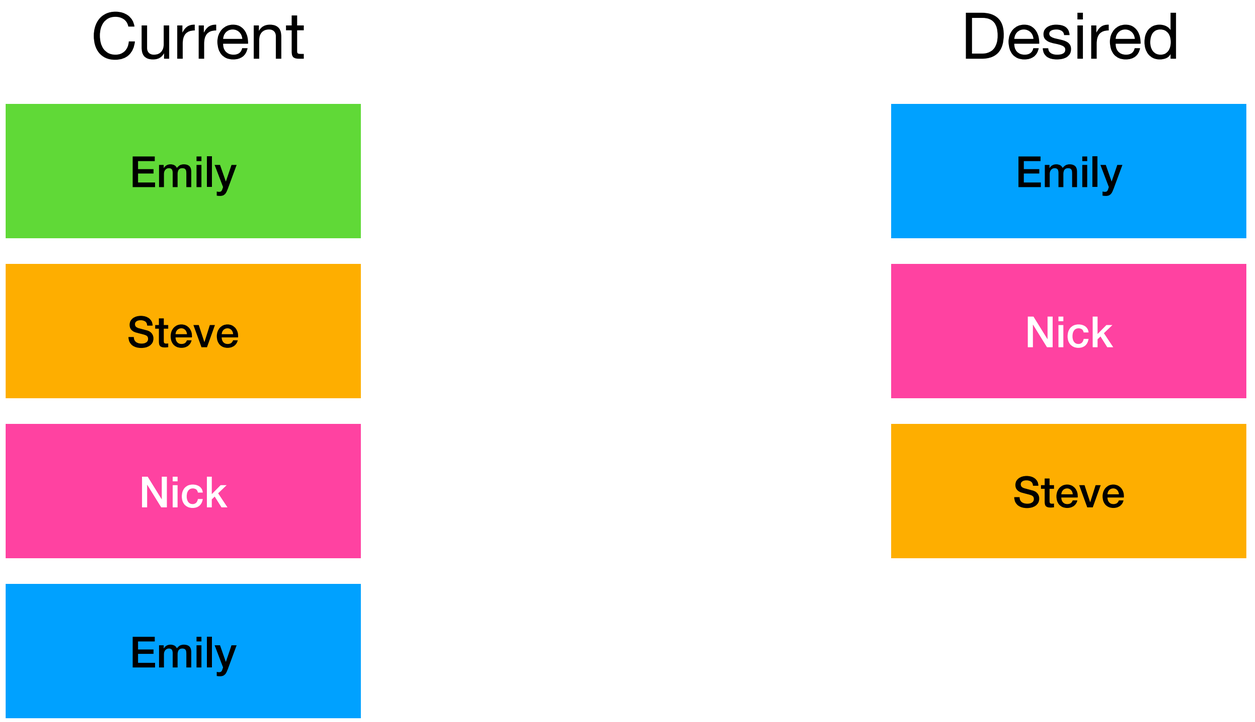
```
<script setup>
const friends = […];
</script>

<template>
<ul>
  <li>

  </li>
</ul>

</template>
```

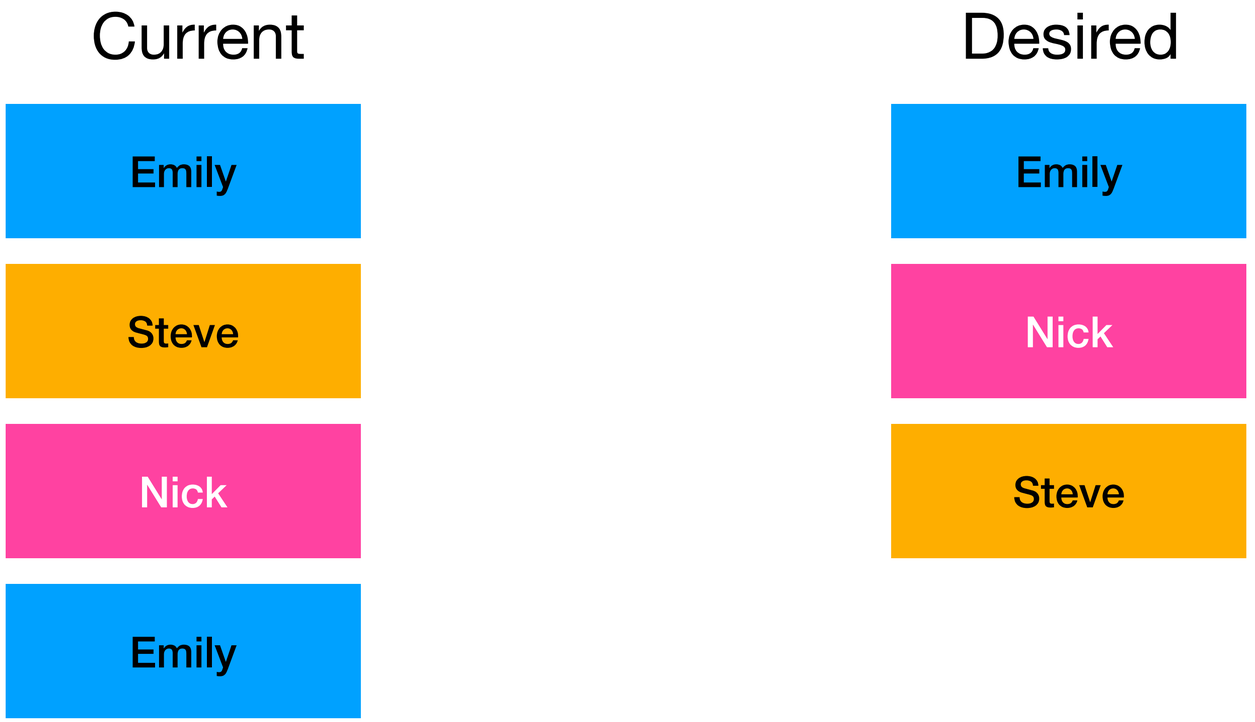
```
<script setup>
const friends = […];
</script>

<template>
<ul>
  <li v-for=“friend in friends”>

  </li>
</ul>

</template>
```

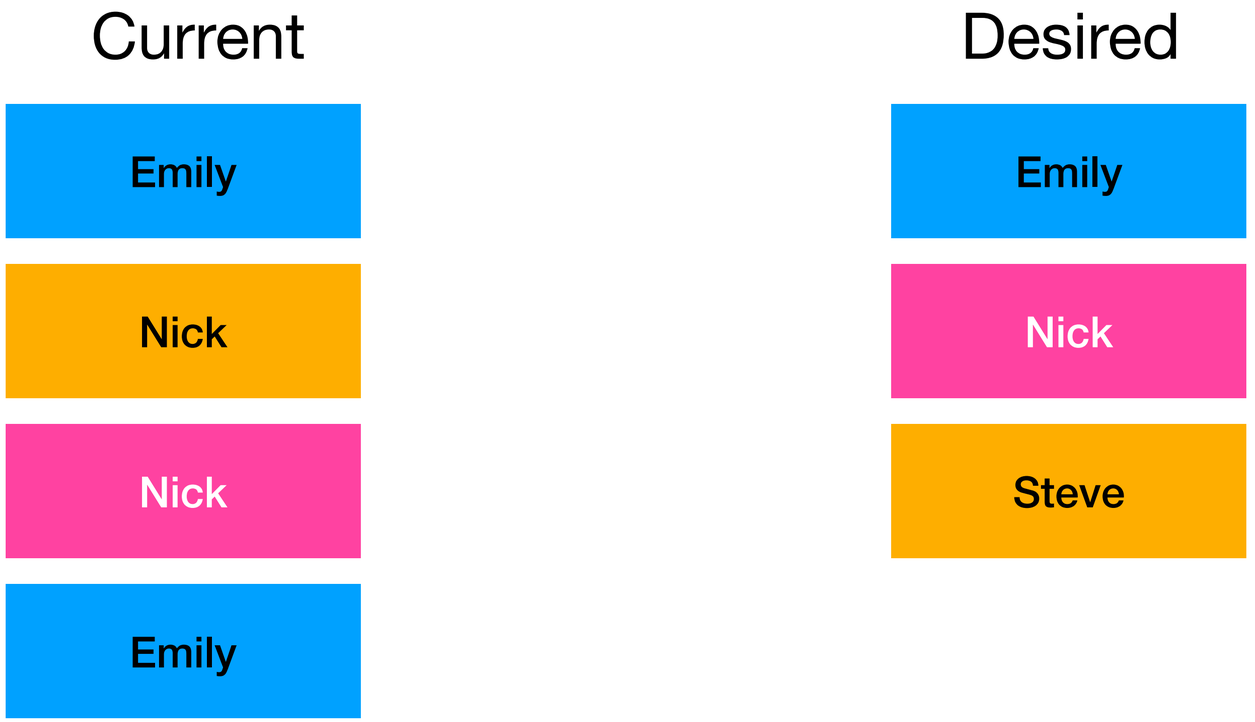
```
<script setup>
const friends = […];
</script>

<template>
<ul>
  <li v-for=“friend in friends”>
    {{ friend.name }}
  </li>
</ul>

</template>
```

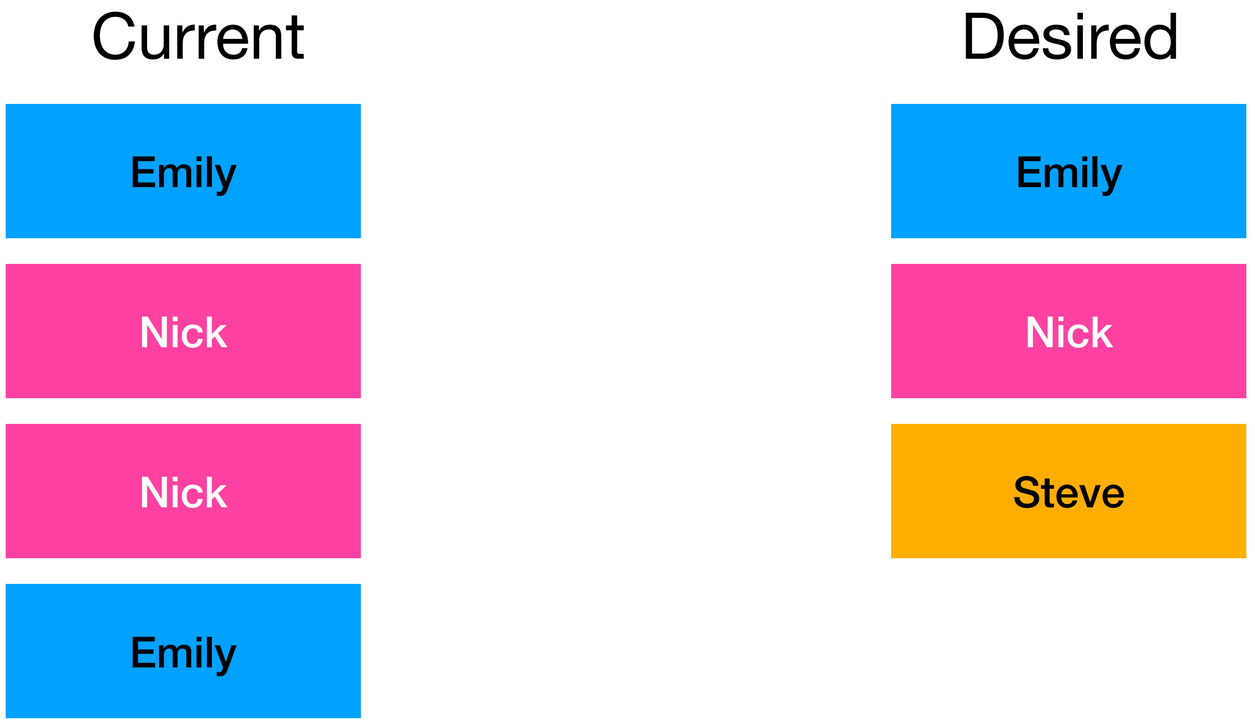
```
<script setup>
const friends = […];
</script>

<template>
<ul>
  <li v-for=“friend in friends” :key=“friend.id">
    {{ friend.name }}
  </li>
</ul>

</template>
```

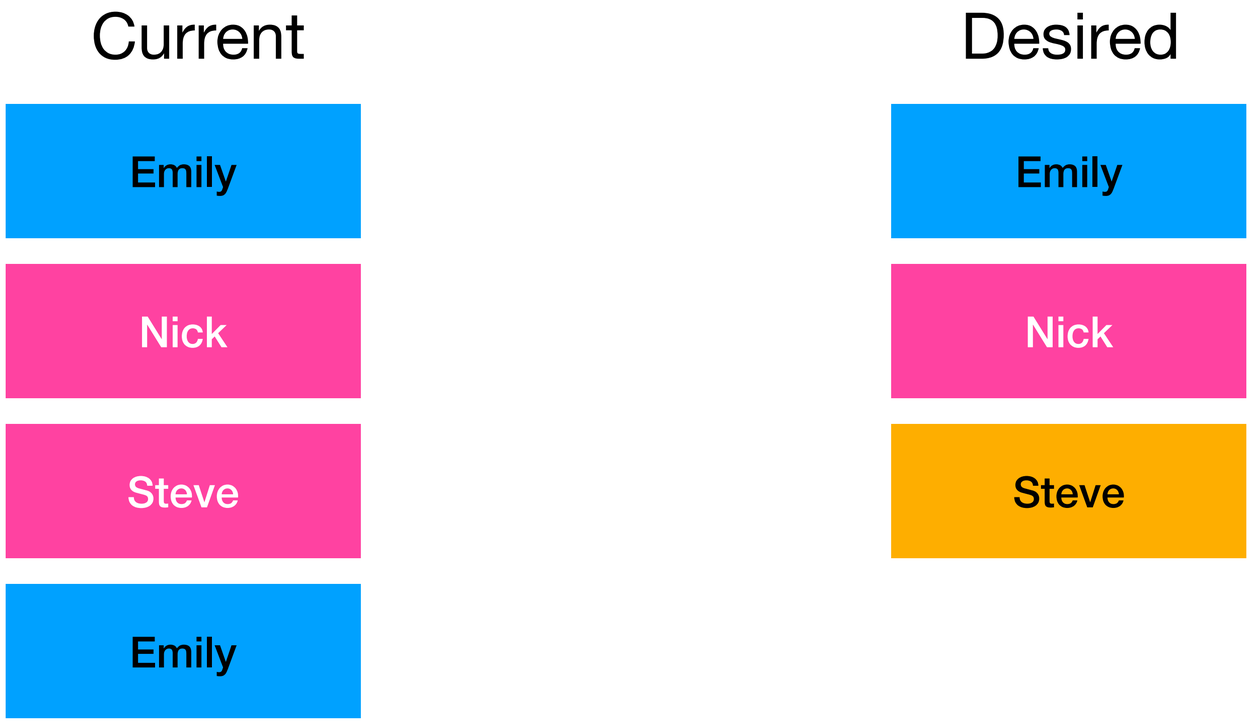

Current
Seth
Steve
Nick
Emily
Desired
Emily
Nick
Steve

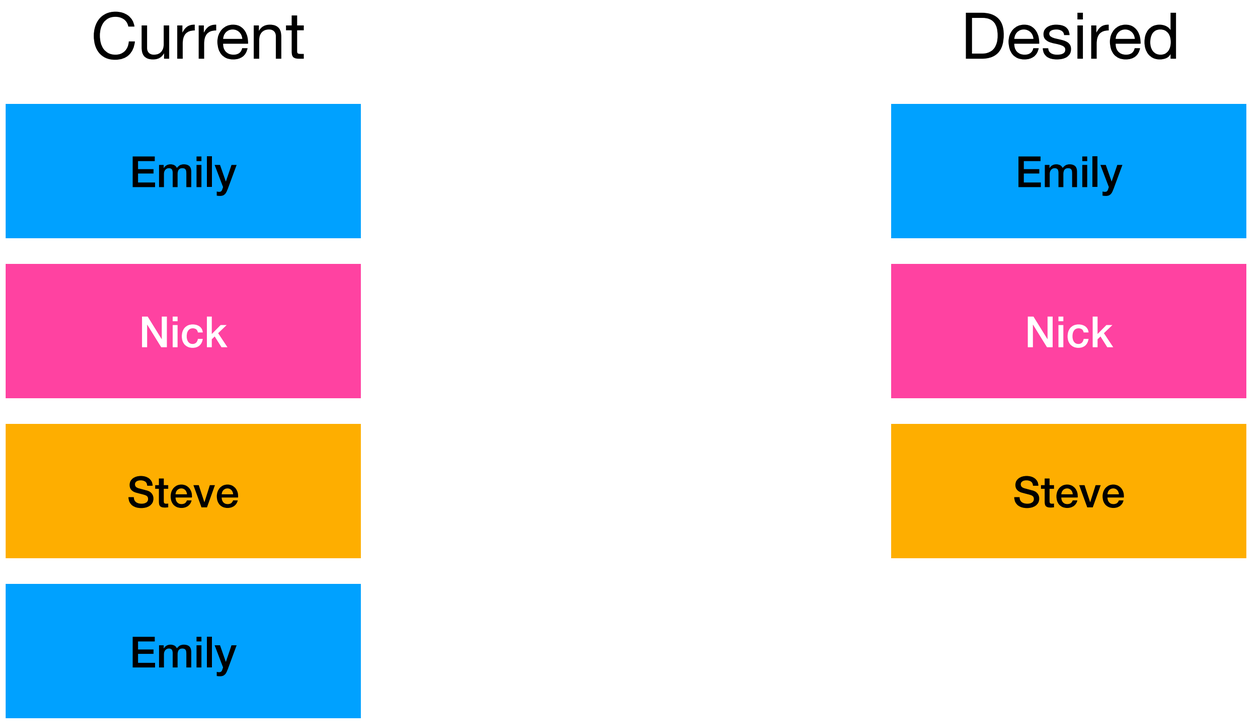

Current
Emily
Steve
Nick
Emily
Desired
Emily
Nick
Steve

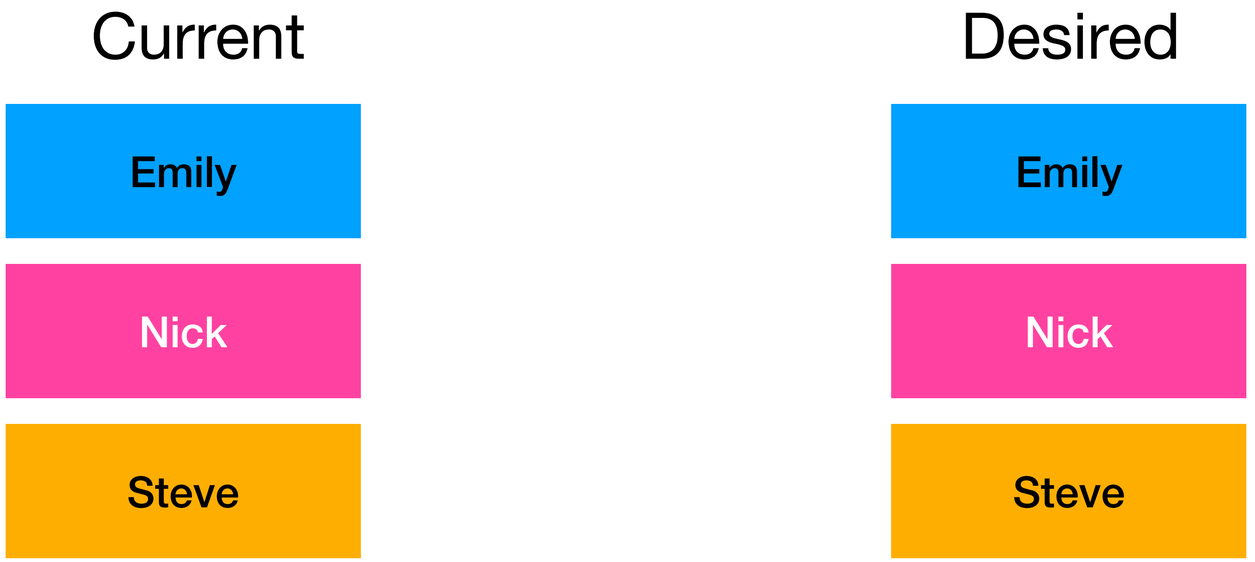

Current
Emily
Steve
Nick
Emily
Desired
Emily
Nick
Steve


Current
Emily
Nick
Nick
Emily
Desired
Emily
Nick
Steve

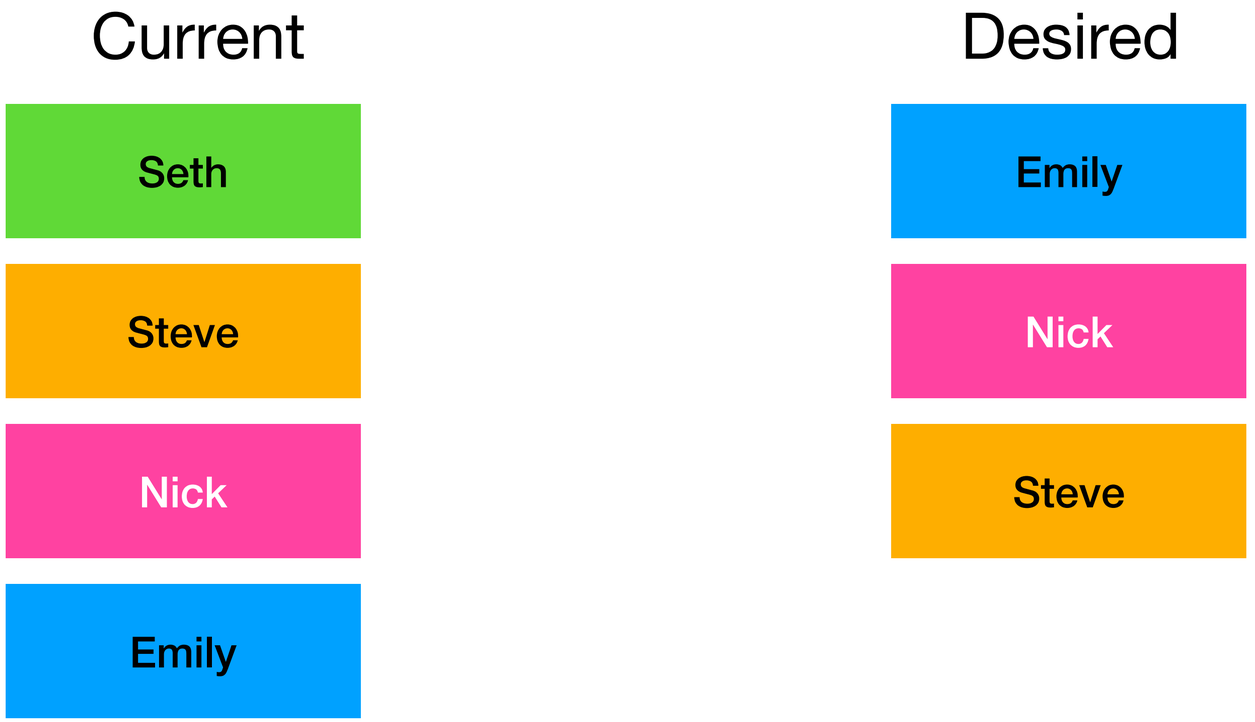

Current
Emily
Nick
Nick
Emily
Desired
Emily
Nick
Steve

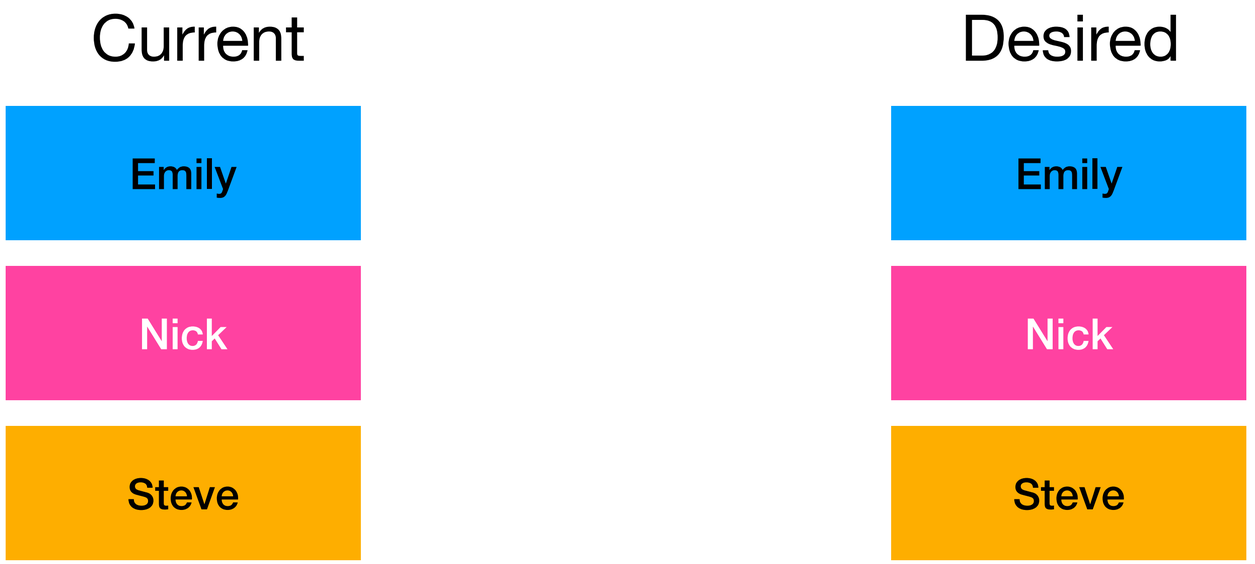

Current
Emily
Nick
Steve
Emily
Desired
Emily
Nick
Steve


Current
Emily
Nick
Steve
Emily
Desired
Emily
Nick
Steve


Current
Emily
Nick
Steve
Desired
Emily
Nick
Steve

With :key


Current
Seth
Steve
Nick
Emily
Desired
Emily
Nick
Steve


Current
Emily
Nick
Steve
Desired
Emily
Nick
Steve

https://www.mattburkedev.com/vue-workshop/templating/

Forms & Reactivity

Reactive

Ref

```
<script setup>
const counter = ref(0);

function increment() {
  counter.value++;
}
</script>

<template>
<button @click="increment">
  Clicked {{ counter }} times
</button>
</template>
```

```
<script setup>
const counter = ref(0);

function increment() {
  counter.value++;
}
</script>

<template>
<button @click="increment">
  Clicked {{ counter }} times
</button>
</template>
```

```
<script setup>
const counter = ref(0);

function increment() {
  counter.value++;
}
</script>

<template>
<button @click="increment">
  Clicked {{ counter }} times
</button>
</template>
```

```
<script setup>
const counter = ref(0);

function increment() {
  counter.value++;
}
</script>

<template>
<button @click="increment">
  Clicked {{ counter }} times
</button>
</template>
```

```
<script setup>
const counter = ref(0);

function increment() {
  counter.value++;
}
</script>

<template>
<button @click="increment">
  Clicked {{ counter }} times
</button>
</template>
```

```
<script setup>
const counter = ref(0);

function increment() {
  counter.value++;
}
</script>

<template>
<button @click="increment">
  Clicked {{ counter }} times
</button>
</template>
```

```
<script setup>
const counter = ref(0);

function increment() {
  counter.value++;
}
</script>

<template>
<button @click="increment">
  Clicked {{ counter }} times
</button>
</template>
```

Reactive

```
<script setup>
const username = ref('');
const email = ref('');



function setEmail(newEmail) {
  email.value = newEmail;
}
</script>

<template>
<p>Username: {{ username }}</p>
<p>Email: {{ email }}</p>
</template>
```

```
<script setup>
const username = ref('');
const email = ref('');



function setEmail(newEmail) {
  email.value = newEmail;
}
</script>

<template>
<p>Username: {{ username }}</p>
<p>Email: {{ email }}</p>
</template>
```

```
<script setup>
const state = reactive({
  username: '',
  email: ''
});

function setEmail(newEmail) {
  email.value = newEmail;
}
</script>

<template>
<p>Username: {{ username }}</p>
<p>Email: {{ email }}</p>
</template>
```

```
<script setup>
const state = reactive({
  username: '',
  email: ''
});

function setEmail(newEmail) {
  email.value = newEmail;
}
</script>

<template>
<p>Username: {{ username }}</p>
<p>Email: {{ email }}</p>
</template>
```

```
<script setup>
const state = reactive({
  username: '',
  email: ''
});

function setEmail(newEmail) {
  state.email = newEmail;
}
</script>

<template>
<p>Username: {{ username }}</p>
<p>Email: {{ email }}</p>
</template>
```

```
<script setup>
const state = reactive({
  username: '',
  email: ''
});

function setEmail(newEmail) {
  state.email = newEmail;
}
</script>

<template>
<p>Username: {{ username }}</p>
<p>Email: {{ email }}</p>
</template>
```

```
<script setup>
const state = reactive({
  username: '',
  email: ''
});

function setEmail(newEmail) {
  state.email = newEmail;
}
</script>

<template>
<p>Username: {{ state.username }}</p>
<p>Email: {{ state.email }}</p>
</template>
```

Computed

```
<script setup>
const firstName = ref('Matt');
const lastName = ref('Burke');

const fullName = computed(() => {
  return firstName.value + lastName.value;
});
</script>

<template>
<p>Hello, {{ fullName }}</p>
</template>
```

```
<script setup>
const firstName = ref('Matt');
const lastName = ref('Burke');

const fullName = computed(() => {
  return firstName.value + lastName.value;
});
</script>

<template>
<p>Hello, {{ fullName }}</p>
</template>
```

Model Binding

```
<script setup>
const text = ref('');
</script>

<template>




</template>
```

```
<script setup>
const text = ref('');
</script>

<template>
<input
  :value="text"
  @input="(event) ⇒ text = event.target.value"
/>
</template>
```

```
<script setup>
const text = ref('');
</script>

<template>
<input
  :value="text"
  @input="(event) ⇒ text = event.target.value"
/>
</template>
```

```
<script setup>
const text = ref('');
</script>

<template>
<input
  :value="text"
  @input="(event) ⇒ text = event.target.value"
/>
</template>
```

```
<script setup>
const text = ref('');
</script>

<template>
<input
  v-model="text"

/>
</template>
```

```
<input v-model="text" />
```

```
<textarea v-model=“text”>
</textarea>
```

```
<input type="checkbox"
v-model="checked" />
```

```
<input type=“radio” value=“on”  v-model=“picked” />
<input type=“radio” value=“off” v-model=“picked” />
```

```
<input type=“radio” value=“on”  v-model=“picked” />
<input type=“radio” value=“off” v-model=“picked” />
```

```
<input type=“radio” value=“on”  v-model=“picked” />
<input type=“radio” value=“off” v-model=“picked” />
```

```
<select v-model="selected">
  <option value="OH">Ohio</option>
  <option value="MO">Missouri</option>
  <option value="IA">Iowa</option>
  <option value="PA">Pennsylvania</option>
  …
</select>
```

https://www.mattburkedev.com/vue-workshop/forms--reactivity/

Components

defineProps

```
<script setup lang=“ts”>
interface WelcomeProps {
  firstName: string,
  message: string
}

definePropsWelcomeProps>();
</script>

<template>
<p>Welcome {{ firstName }}</p>
<p>{{ message }}</p>
</template>
```

```
<script setup lang="ts">
interface WelcomeProps {
  firstName: string,
  message: string
}

defineProps<WelcomeProps>();
</script>

<template>
<p>Welcome {{ firstName }}</p>
<p>{{ message }}</p>
</template>
```

```
<script setup lang="ts">
interface WelcomeProps {
  firstName: string,
  message: string
}

defineProps<WelcomeProps>();
</script>

<template>
<p>Welcome {{ firstName }}</p>
<p>{{ message }}</p>
</template>
```

```
<script setup lang="ts">
interface WelcomeProps {
  firstName: string,
  message: string
}

defineProps<WelcomeProps>();
</script>

<template>
<p>Welcome {{ firstName }}</p>
<p>{{ message }}</p>
</template>
```

```
<script setup>
import WelcomeMsg from ‘./WelcomeMsg.vue’;

const firstName = ref(‘’);
</script>

<template>
<WelcomeMsg
  :first-name=“firstName”
  message=“Welcome to CodeMash 2024”
/>
</template>
```

```
<script setup>
import WelcomeMsg from ‘./WelcomeMsg.vue’;

const firstName = ref(‘’);
</script>

<template>
<WelcomeMsg
  :first-name=“firstName”
  message=“Welcome to CodeMash 2024”
/>
</template>
```

```
<script setup>
import WelcomeMsg from ‘./WelcomeMsg.vue’;

const firstName = ref(‘’);
</script>

<template>
<WelcomeMsg
  :first-name=“firstName”
  message=“Welcome to CodeMash 2024”
/>
</template>
```

defineEmits

```
<script setup>
const emit = defineEmits([“login”]);









</script>

<template>
<form>




</form>

</template>
```

```
<script setup>
const emit = defineEmits([“login”]);

const state = reactive({ username: ‘’, password: ‘’  });




</script>

<template>
<form>
  <input type=“text” v-model=“state.username” placeholder=“Username” />
  <input type=“password” v-model=“state.password” placeholder=“password” />

</form>

</template>
```

```
<script setup>
const emit = defineEmits([“login”]);

const state = reactive({ username: ‘’, password: ‘’  });

function handleSubmit() {
  emit(‘login’, state);
}
</script>

<template>
<form>
  <input type=“text” v-model=“state.username” placeholder=“Username” />
  <input type=“password” v-model=“state.password” placeholder=“password” />
  <input type=“submit” @click=“handleSubmit” />
</form>

</template>
```

```
<script setup>
const emit = defineEmits([“login”]);

const state = reactive({ username: ‘’, password: ‘’  });

function handleSubmit() {
  emit(‘login’, state);
}
</script>

<template>
<form>
  <input type=“text” v-model=“state.username” placeholder=“Username” />
  <input type=“password” v-model=“state.password” placeholder=“password” />
  <input type=“submit” @click=“handleSubmit” />
</form>

</template>
```

```
<script setup>
const emit = defineEmits([“login”]);

const state = reactive({ username: ‘’, password: ‘’  });

function handleSubmit() {
  emit(‘login’, state);
}
</script>

<template>
<form>
  <input type=“text” v-model=“state.username” placeholder=“Username” />
  <input type=“password” v-model=“state.password” placeholder=“password” />
  <input type=“submit” @click=“handleSubmit” />
</form>

</template>
```

Handling Component Events

```
<script setup>




</script>

<template>

</template>
```

```
<script setup>
import LoginForm from ‘./LoginForm.vue’;




</script>

<template>

</template>
```

```
<script setup>
import LoginForm from ‘./LoginForm.vue’;




</script>

<template>
<LoginForm @login=“handleLogin” />
</template>
```

```
<script setup>
import LoginForm from ‘./LoginForm.vue’;




</script>

<template>
<LoginForm @login=“handleLogin” />
</template>
```

```
<script setup>
import LoginForm from ‘./LoginForm.vue’;




</script>

<template>
<LoginForm @login=“handleLogin” />
</template>
```

```
<script setup>
import LoginForm from ‘./LoginForm.vue’;

async function handleLogin(data) {
  await api.login(data);
}
</script>

<template>
<LoginForm @login=“handleLogin” />
</template>
```

```
<script setup>
import LoginForm from ‘./LoginForm.vue’;

async function handleLogin(data) {
  await api.login(data);
}
</script>

<template>
<LoginForm @login=“handleLogin” />
</template>
```

```
<script setup>
import LoginForm from ‘./LoginForm.vue’;

async function handleLogin(data) {
  await api.login(data);
}
</script>

<template>
<LoginForm @login=“handleLogin” />
</template>
```

Type Safe Emit Function

```
const emit = defineEmits({
  (e: ‘change’, id: number): void,
  (e: ‘update’, value: string): void
});

emit(‘change’, 42);
emit(‘update’, ‘foo’);
```

```
const emit = defineEmits({
  (e: ‘change’, id: number): void,
  (e: ‘update’, value: string): void
});

emit(‘change’, 42);
emit(‘update’, ‘foo’);
```

```ts
const emit = defineEmits({
  (e: ‘change’, id: number): void,
  (e: ‘update’, value: string): void
});

emit(‘change’, 42);
emit(‘update’, ‘foo’);
```

```
const emit = defineEmits({
  (e: ‘change’, id: number): void,
  (e: ‘update’, value: string): void
});

emit(‘change’, 42);
emit(‘update’, ‘foo’);
```

```
const emit = defineEmits({
  (e: ‘change’, id: number): void,
  (e: ‘update’, value: string): void
});

emit(‘change’, 42);
emit(‘update’, ‘foo’);
```

```
const emit = defineEmits({
  (e: ‘change’, id: number): void,
  (e: ‘update’, value: string): void
});

emit(‘change’, 42);
emit(‘update’, ‘foo’);
emit(‘updtae’, ‘bar’);
```

```
src/components/SessionList.vue:22:6 - error TS2769: No overload matches this call.
  Overload 1 of 2, '(e: "change", value: number): void', gave the following error.
    Argument of type '"updtae"' is not assignable to parameter of type '"change"'.
  Overload 2 of 2, '(e: "update", value: string): void', gave the following error.
    Argument of type '"updtae"' is not assignable to parameter of type '"update"'.

22 emit('updtae', 'bar');
       ~~~~~~~~
```

```
const emit = defineEmits({
})

emit('updtae', 'bar');
```

Component V-Model

```
<script setup lang=“ts”>




</script>
```

```
<script setup lang=“ts”>
definePropss<{ modelValue: string }>();









</script>
```

```
<script setup lang="ts">
defineProps<{ modelValue: string }>();









</script>
```

```
<script setup lang=“ts”>
definePropss<{ modelValue: string }>();

const emit = defineEmits<{
  (e: ‘update:modelValue’, value: string): void
}>();




</script>
```

```
<script setup lang="ts">
definePropss<{ modelValue: string }>();

const emit = defineEmits<{
  (e: 'update:modelValue', value: string): void
}>();




</script>
```

```
<script setup lang="ts">
definePropsstring }>();

const emit = defineEmits<{
  (e: 'update:modelValue', value: string): void
}>();




</script>
```

```
<script setup lang=“ts”>
definePropss<{ modelValue: string }>();

const emit = defineEmits<{
  (e: ‘update:modelValue’, value: string): void
}>();

function onInput(e: KeyboardEvent) {
  emit(‘update:modelValue’, e.target.value);
}
</script>
```

```
<script setup lang="ts">
defineProps<{ modelValue: string }>();

const emit = defineEmits<{
  (e: 'update:modelValue', value: string): void
}>();

function onInput(e: KeyboardEvent) {
  emit('update:modelValue', e.target.value);
}
</script>
```

```
<script setup>
const text = ref(‘’);
</script>

<template>
<FancyTextBox


/>
</template>
```

```
<script setup>
const text = ref(‘’);
</script>

<template>
<FancyTextBox
  :model-value=“text”

/>
</template>
```

```
<script setup>
const text = ref(‘’);
</script>

<template>
<FancyTextBox
  :model-value=“text”

/>
</template>
```

```
<script setup>
const text = ref(‘’);
</script>

<template>
<FancyTextBox
  :model-value=“text”

/>
</template>
```

```
<script setup>
const text = ref(‘’);
</script>

<template>
<FancyTextBox
  :model-value=“text”
  @update:modelValue=“(value) ⇒ text = value”
/>
</template>
```

```
<script setup>
const text = ref('');
</script>

<template>
<FancyTextBox
  :model-value="text"
  @update:modelValue="(value) => text = value"
/>
</template>
```

```
<script setup>
const text = ref('');
</script>

<template>
<FancyTextBox
  :model-value="text"
  @update:modelValue="(value) => text = value"
/>
</template>
```

```
<script setup>
const text = ref('');
</script>

<template>
<FancyTextBox
  :model-value="text"
  @update:modelValue="(value) => text = value"
/>
</template>
```

```
<script setup>
const text = ref(‘’);
</script>

<template>
<FancyTextBox
  v-model=“text”

/>
</template>
```

Props Down, Events Up

https://www.mattburkedev.com/vue-workshop/components/

Routing

Routing matches URLs to Components

/	HomePage
/sessions	SessionList
/session/:id	SessionDetails

/	HomePage
/sessions	SessionList
/session/:id	SessionDetails

Vue Router
The official Router for Vue.js

Expressive, configurable and convenient routing for Vue.js

Get Started

Free Video Course

Get the Vue Router Cheat Sheet

Expressive route syntax

Define static and dynamic routes with an intuitive and powerful syntax.

Fine-grained Navigation control

Intercept any navigation and precisely control its outcome.

Component-based configuration

Map each route to the component that should display.

History modes

Choose between HTML5, Hash or Memory

Scroll control

Precisely control the scroll position in every

Automatic Encoding

Directly use unicode characters (你好) in your

<router-view />

Sessions

Search

Track
All

Introductory and overview | Intermediate | Advanced

Showing 211 of 211 talks

Intermediate

Build your own AI sidekick with Azure AI, Semantic Kernel, and .NET 8

Matt Eland, Samuel Gomez

Are you a budding super hero or super villain* but you feel like you're lacking that AI companion to help you reach that next level? Have you ever wanted to see what AI can do to help your daily life? Do you have an interesting application that would just be so much better with a little artificial...

Tuesday 8:00 - 12:00 Details

Intermediate

Building Applications on Top of Large Language Models (LLMs) (Part 1)

Nilanjan Raychaudhuri, BJ Allmon

It's no secret that a new generation of powerful and highly-scaled language models is taking the world by storm. Large language models are becoming a powerful new primitive for building software. In this precompiler, we will deep dive into the rapidly evolving landscape of Large Language Models...

Tuesday 8:00 - 12:00 Details

Intermediate

Mastering Solutions Architecture with Design Katas (Part 1)

Gaines Kergosien, Eric Potter

TLDR: Architects get relatively few opportunities to practice their craft, so we will group up to formulate architectural visions for "real world" business problems. Attendees will then evaluate each group's solution to gain insight into the pros and cons of different approaches. Fred Brooks said,...

Tuesday 8:00 - 12:00 Details

Introductory and overview

Build a Modern Single Page Application with Vue

Matt Burke

Vue offers developers a way to build ambitious front-end applications with powerful reactive programming patterns and an intuitive HTML-based templating language. This workshop will give you a jumping-off point for large front-end applications using Vue with blazing-fast dev tools like esbuild,...

Tuesday 8:00 - 12:00 Details

Intermediate

Hands on with OpenTelemetry

Nočnica Mellifera

Introductory and overview

What is Your Working Genius? (Workshop)

Introductory and overview

Accessibility Auditing: Getting Started with Accessibility (Part

Introductory and overview

Building Trust and Breaking Barriers (with LEGO!)

Sessions

Search

Track
All

Introductory and overview | Intermediate | Advanced

Showing 211 of 211 talks

Intermediate

Build your own AI sidekick with Azure AI, Semantic Kernel, and .NET 8

Matt Eland, Samuel Gomez

Are you a budding super hero or super villain* but you feel like you're lacking that AI companion to help you reach that next level? Have you ever wanted to see what AI can do to help your daily life? Do you have an interesting application that would just be so much better with a little artificial...

Tuesday 8:00 - 12:00 Details

Intermediate

Building Applications on Top of Large Language Models (LLMs) (Part 1)

Nilanjan Raychaudhuri, BJ Allmon

It's no secret that a new generation of powerful and highly-scaled language models is taking the world by storm. Large language models are becoming a powerful new primitive for building software. In this precompiler, we will deep dive into the rapidly evolving landscape of Large Language Models...

Tuesday 8:00 - 12:00 Details

Intermediate

Mastering Solutions Architecture with Design Katas (Part 1)

Gaines Kergosien, Eric Potter

TLDR: Architects get relatively few opportunities to practice their craft, so we will group up to formulate architectural visions for "real world" business problems. Attendees will then evaluate each group's solution to gain insight into the pros and cons of different approaches. Fred Brooks said,...

Tuesday 8:00 - 12:00 Details

Introductory and overview

Build a Modern Single Page Application with Vue

Matt Burke

Vue offers developers a way to build ambitious front-end applications with powerful reactive programming patterns and an intuitive HTML-based templating language. This workshop will give you a jumping-off point for large front-end applications using Vue with blazing-fast dev tools like esbuild,...

Tuesday 8:00 - 12:00 Details

Intermediate

Hands on with OpenTelemetry

Nočnica Mellifera

Introductory and overview

What is Your Working Genius? (Workshop)

Introductory and overview

Accessibility Auditing: Getting Started with Accessibility (Part

Introductory and overview

Building Trust and Breaking Barriers (with LEGO!)

<router-view />

Sessions

Search

Track
All

Introductory and overview | Intermediate | Advanced

Showing 211 of 211 talks

Intermediate

Build your own AI sidekick with Azure AI, Semantic Kernel, and .NET 8

Matt Eland, Samuel Gomez

Are you a budding super hero or super villain* but you feel like you're lacking that AI companion to help you reach that next level? Have you ever wanted to see what AI can do to help your daily life? Do you have an interesting application that would just be so much better with a little artificial...

Tuesday 8:00 - 12:00 Details

Intermediate

Building Applications on Top of Large Language Models (LLMs) (Part 1)

Nilanjan Raychaudhuri, BJ Allmon

It's no secret that a new generation of powerful and highly-scaled language models is taking the world by storm. Large language models are becoming a powerful new primitive for building software. In this precompiler, we will deep dive into the rapidly evolving landscape of Large Language Models...

Tuesday 8:00 - 12:00 Details

Intermediate

Mastering Solutions Architecture with Design Katas (Part 1)

Gaines Kergosien, Eric Potter

TLDR: Architects get relatively few opportunities to practice their craft, so we will group up to formulate architectural visions for "real world" business problems. Attendees will then evaluate each group's solution to gain insight into the pros and cons of different approaches. Fred Brooks said,...

Tuesday 8:00 - 12:00 Details

Introductory and overview

Build a Modern Single Page Application with Vue

Matt Burke

Vue offers developers a way to build ambitious front-end applications with powerful reactive programming patterns and an intuitive HTML-based templating language. This workshop will give you a jumping-off point for large front-end applications using Vue with blazing-fast dev tools like esbuild,...

Tuesday 8:00 - 12:00 Details

Intermediate

Hands on with OpenTelemetry

Nočnica Mellifera

Introductory and overview

What is Your Working Genius? (Workshop)

Introductory and overview

Accessibility Auditing: Getting Started with Accessibility (Part

Introductory and overview

Building Trust and Breaking Barriers (with LEGO!)

SessionDetails

```
<router-link>Click Me</router-link>
```

Client-side routing

createWebHashHistory()

http://example.com/#/sessions/1234

- Supports legacy browsers
- Doesn’t require server-side support
- Ugly

createWebHashHistory()

http://example.com/#/sessions/1234

- Supports legacy browsers
- Doesn’t require server-side support
- Ugly

createWebHistory()

http://example.com/sessions/1234

- HTML5 History API
- Requires server-side support

Lazy Loading

```
import(path: string): Promise<any>
```

https://www.mattburkedev.com/vue-workshop/routing/

Data & State Management

https://www.mattburkedev.com/vue-workshop/data--state-management/