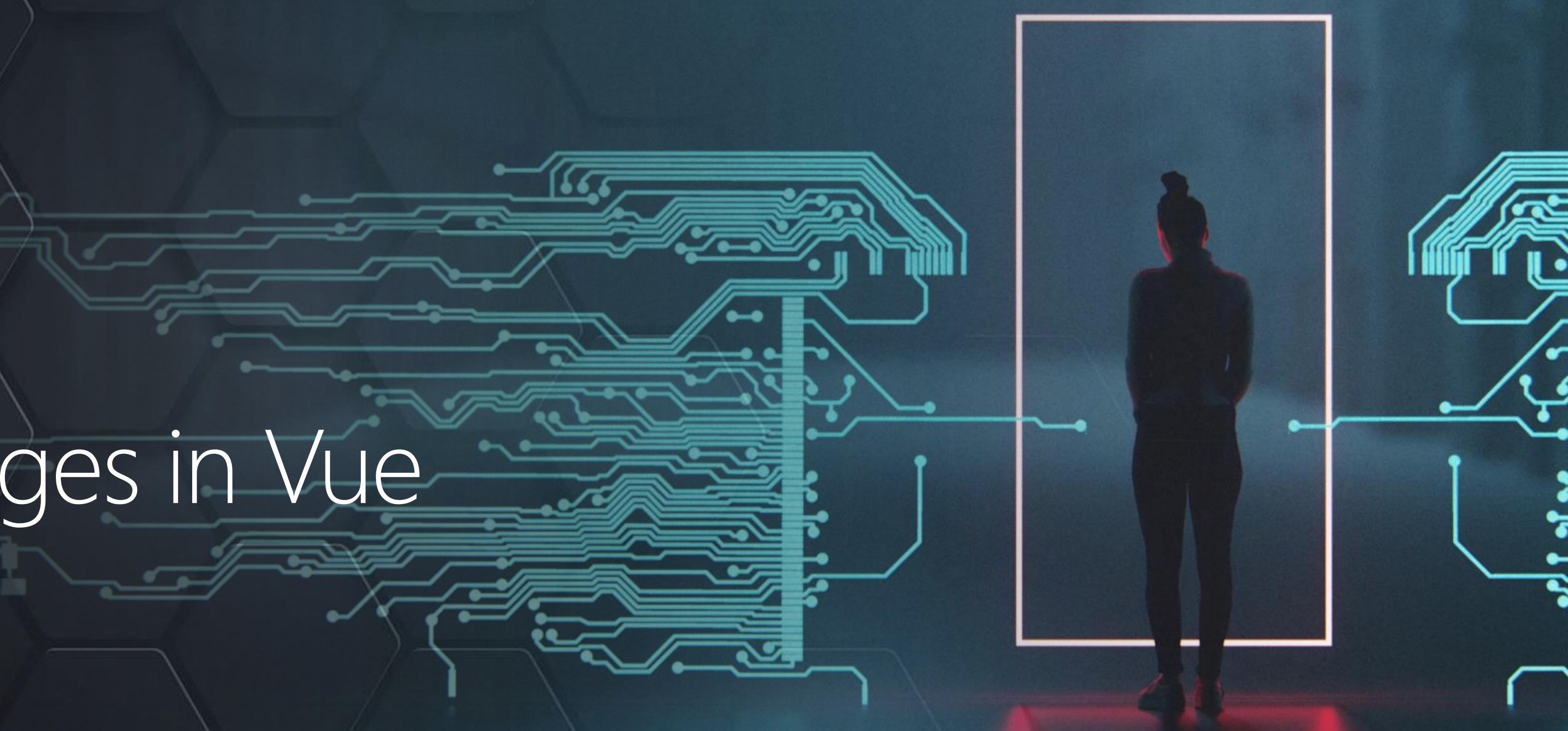
Microsoft

Reactor



Microsoft Reactor
Creating Dynamic Pages in Vue

Event Code: #13101

FAQ's

What are the Reactors?

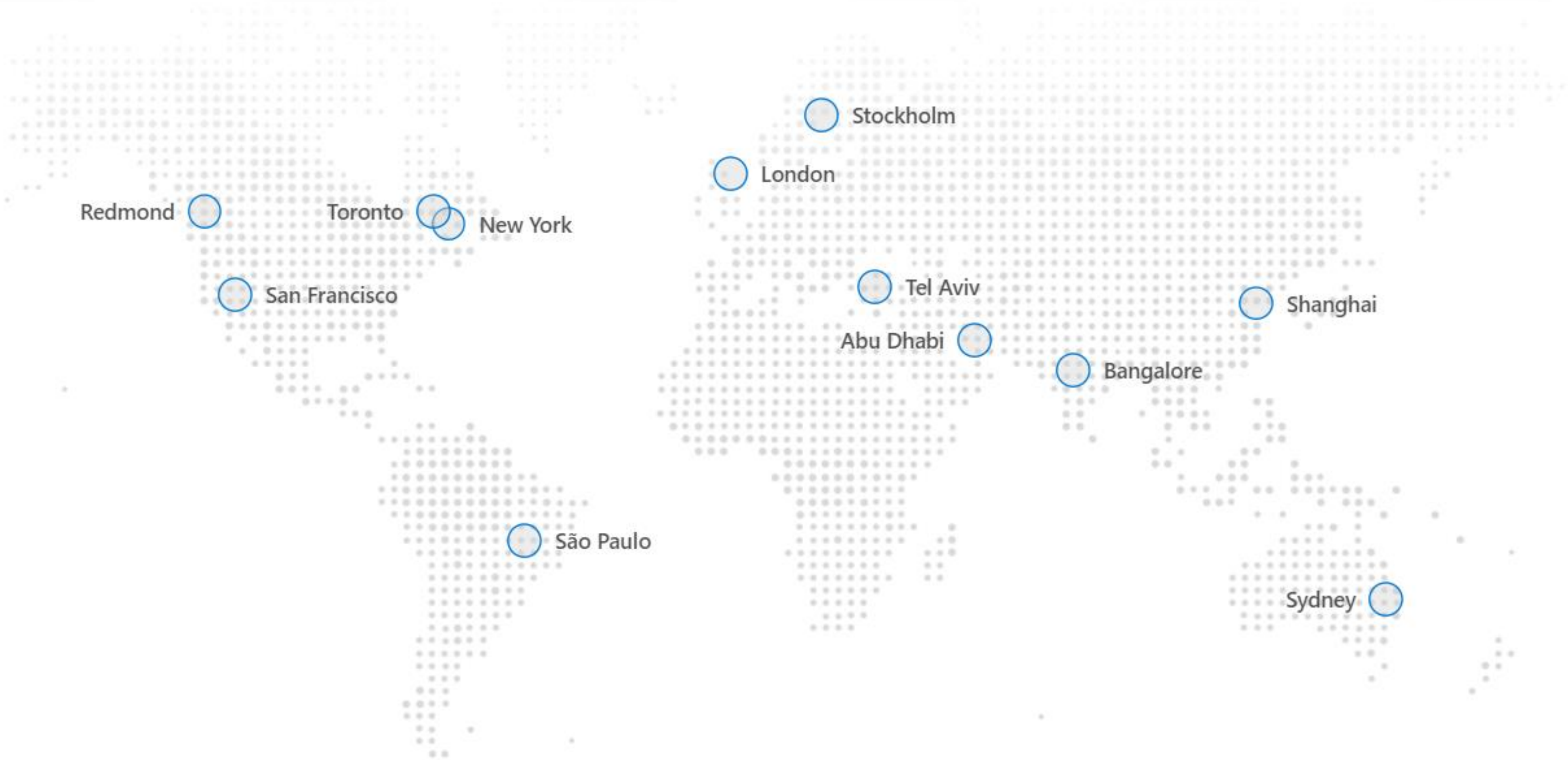
Reactors are community spaces where technology professionals meet, learn, and connect—to both their local peers as well as industry-leading ideas and technology from Microsoft, partners, and the open source community.

With a diverse mix of hands-on workshops, expert panels, and community events, there's something for everyone—whether you're just getting started or working on complex projects.

Reactor programming is always free and inclusive of a broad set of products, tools, and technologies. You will find:

- First party content designed to help developers learn new skills in high-demand fields such as Machine Learning, AI, and Data Science
- New content around breakthrough technologies and concepts such as quantum computing and blockchain
- Resources, talks and classes from Microsoft for Startups to deliver what startups need to succeed
- Cloud Advocates who deep-dive into their technical specialties and the Industry Experiences team to provide tailored content around applications for Manufacturing, Healthcare, Retail, and other verticals
- Reactors always welcome community groups, Meetups, partners and MVPs to use our space

Map


Stockholm
London
Redmond
Toronto
New York
San Francisco
Tel Aviv
Abu Dhabi
Shanghai
Bangalore
São Paulo
Sydney

Our Ask of You

Be aware of others

Be friendly and patient

Be welcoming and respectful

Be open to all questions and viewpoints

Be understanding of differences

Be kind and considerate to others

Creating Dynamic Pages in Vue

Allen Sanders

Principal Cloud Architect

Speaker Bio: Allen Sanders is a Principal Cloud Architect with over 25 years of experience in software engineering, architecture and design delivering technology strategy and business solutions in multiple verticals. As a Microsoft Certified Professional, he has taught, led and mentored teams of varying sizes in multiple application transformation efforts. He has a passion for learning – both his own and for others.

Agenda

1 Introduction

2 Rendering Lists

3 Conditional Rendering

4 Working with Forms

5 Working with Events

6 Working with Computed Properties

7 Knowledge Check

8 Summary

9

10

Introduction

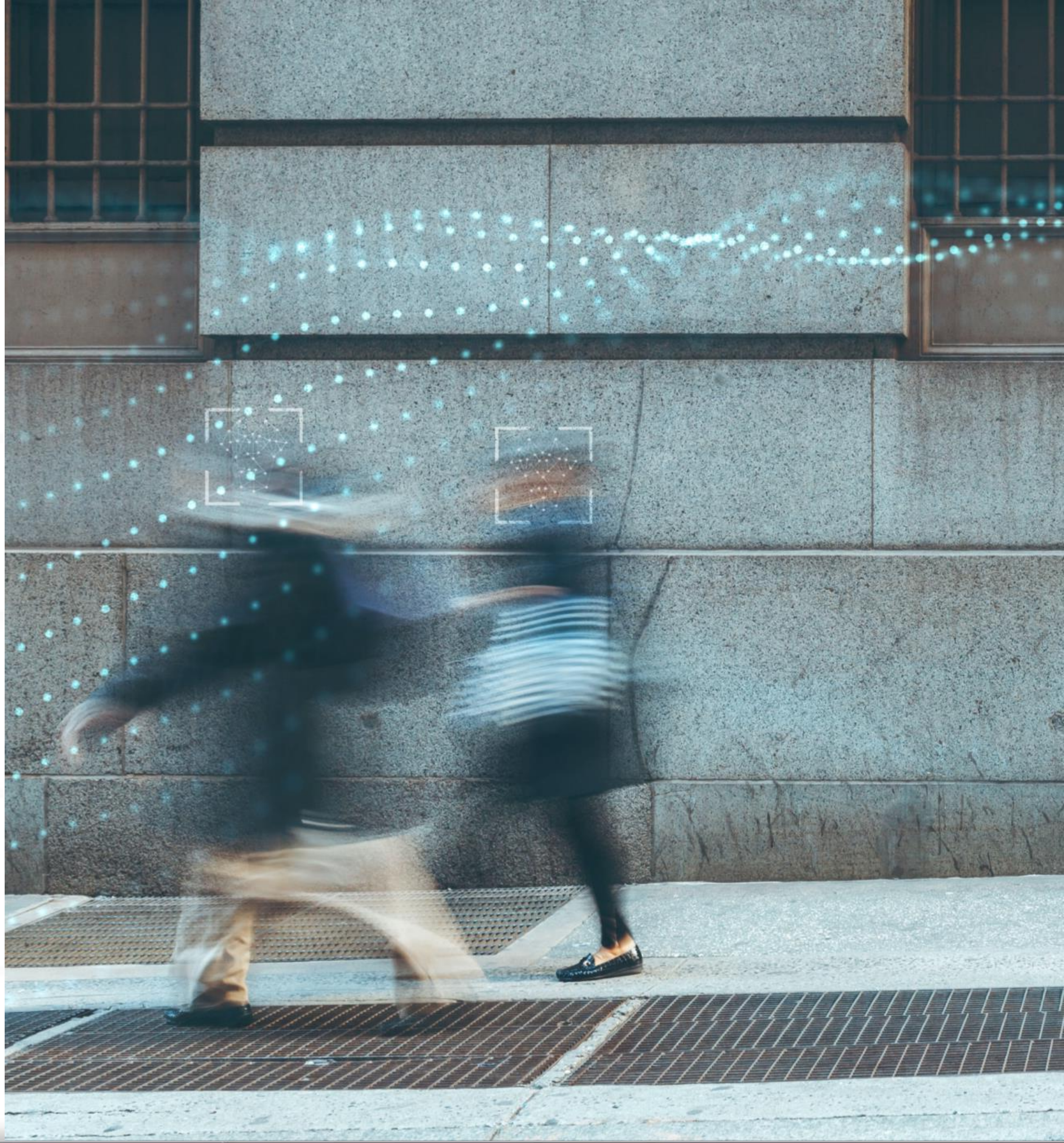
https://aka.ms/VueDynamicRendering

https://aka.ms/VueDataEvents

Reactor

Objectives

- Explore mechanics of building dynamic pages in Vue.js
- Includes dynamic lists and conditional rendering
- Includes forms, events, and computed properties for building responsive UI's



Prerequisites

- HTML and CSS
- JavaScript
- Previous experience with or exposure to Vue.js is helpful

Software Used

- Git
- Visual Studio Code
- Vue.js (see https://vuejs.org/ for more info)

Rendering Lists

v-for

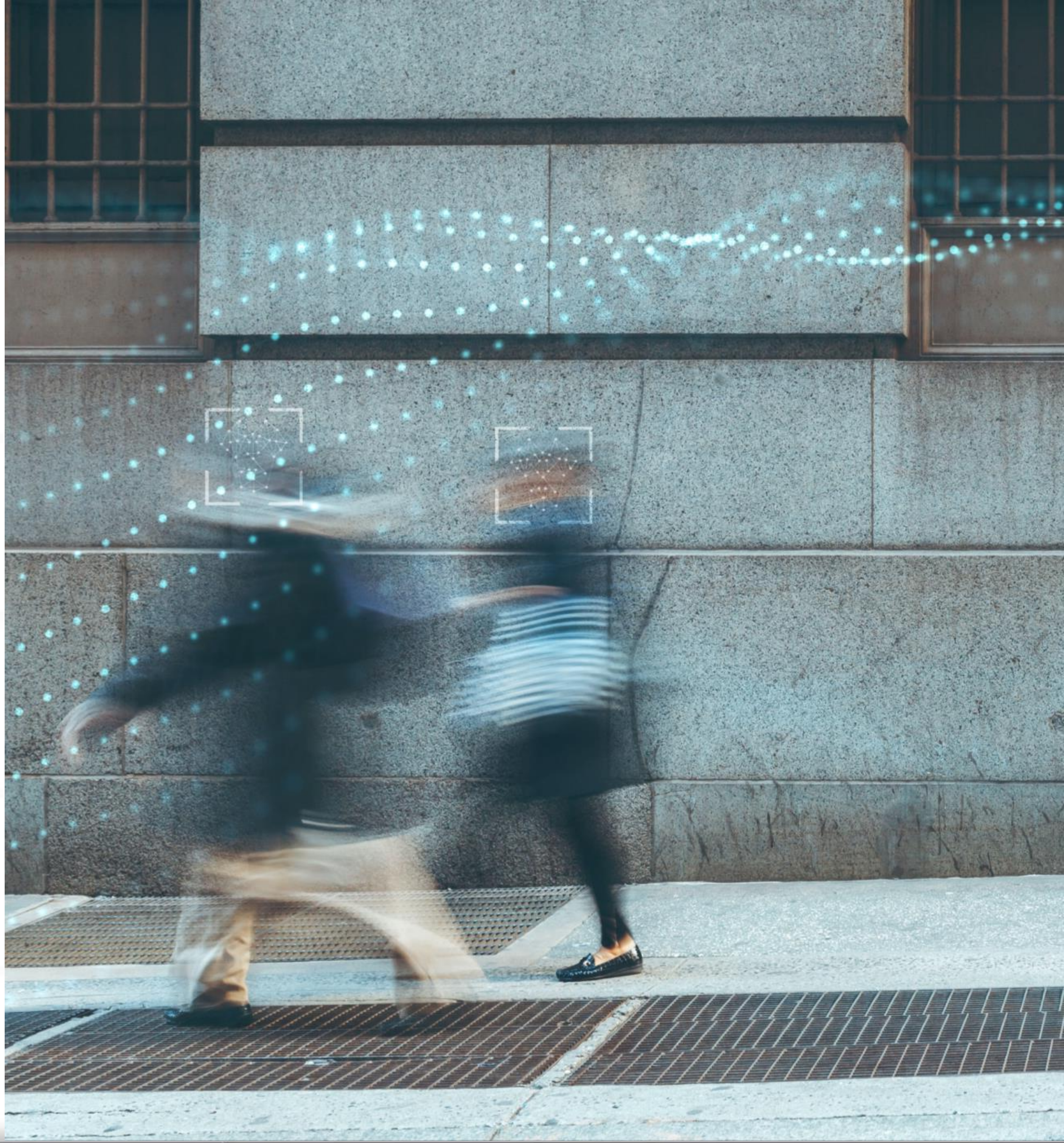
- Can be used in Vue.js to dynamically display a collection
- Vue will render an instance of an element for each collection item
- Will automatically refresh the display on updates to collection

```
<ul id="demo">
    <li v-for="name in names">{{ name }}</div>
</ul>
```

v-for

- With lists (especially large lists), more efficient to update a specific item
- Use the *:key* attribute to specify a unique key for locating the item
- Simplest case uses the numeric index of the collection

```
<ul id="demo">
    <li v-for="(name, index) in names" :key="index">{{ name }}</div>
</ul>
```



Reactor

Conditional Rendering

Conditional Rendering

- Vue.js supports conditional updates to the page
- Available options include:
  - ➢ Toggling visibility
  - ➢ Applying Boolean logic (if/else if/else)

v-show

- Uses a Boolean value or expression to determine visibility
- If true, element will be displayed; if false, it will be hidden
- Element still in DOM – equivalent to using CSS style "*display: none;*"

```
<div v-show="new Date().getMonth() < 3">It is the first quarter</div>
```

v-if, v-else-if, and v-else

- *v-if* uses Boolean value or expression to manage element inclusion
- Can be combined with *v-else-if* and *v-else* for branching
- Key distinction is that if false, *v-if* removes element from DOM

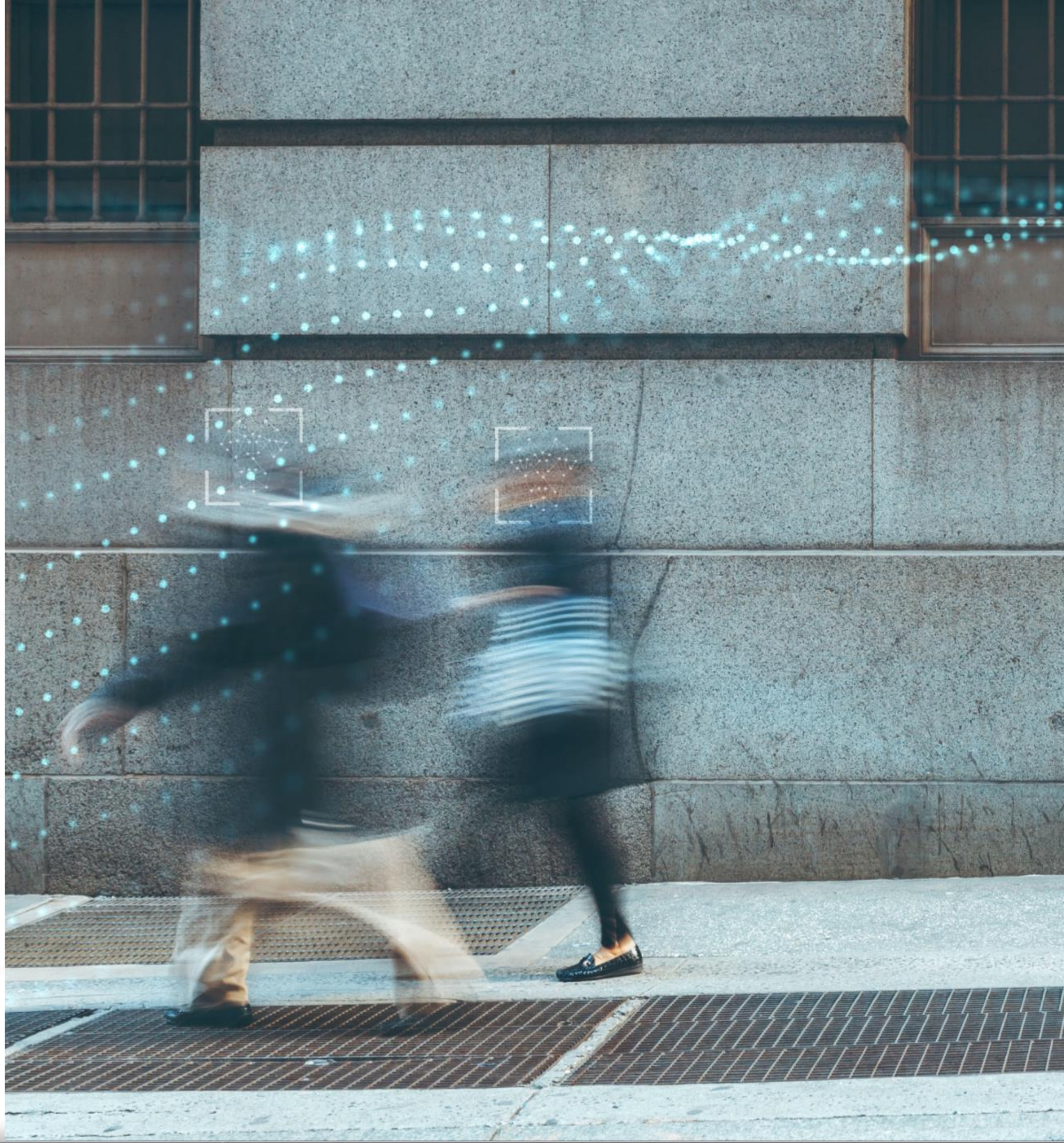
```
<div v-if="new Date().getMonth() < 3">It is the first quarter</div>
<div v-else-if="new Date().getMonth() < 6">It is the second quarter</div>
<div v-else-if="new Date().getMonth() < 9">It is the third quarter</div>
<div v-else>It is the fourth quarter</div>
```

Working with Forms

Microsoft Reactor

Forms and State

- Data returned by *data()* function in app or component is AKA state
- State is usually maintained by users using HTML forms
- *v-bind* is available for dynamically displaying data, but binding is one-way

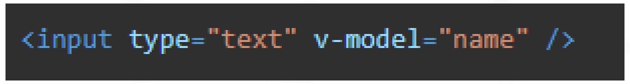


v-model

- Directive used to create two-way binding
- Used to display data but also accepts updates from user in form controls
- Can be used for binding to textboxes, checkboxes, and drop-down lists

v-model with Textboxes

- Applied to input element and references property in data model
- General format is <*input type="text" v-model="<model prop>" />*
- Changes made to text in textbox will be applied to state



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

v-model with Checkboxes

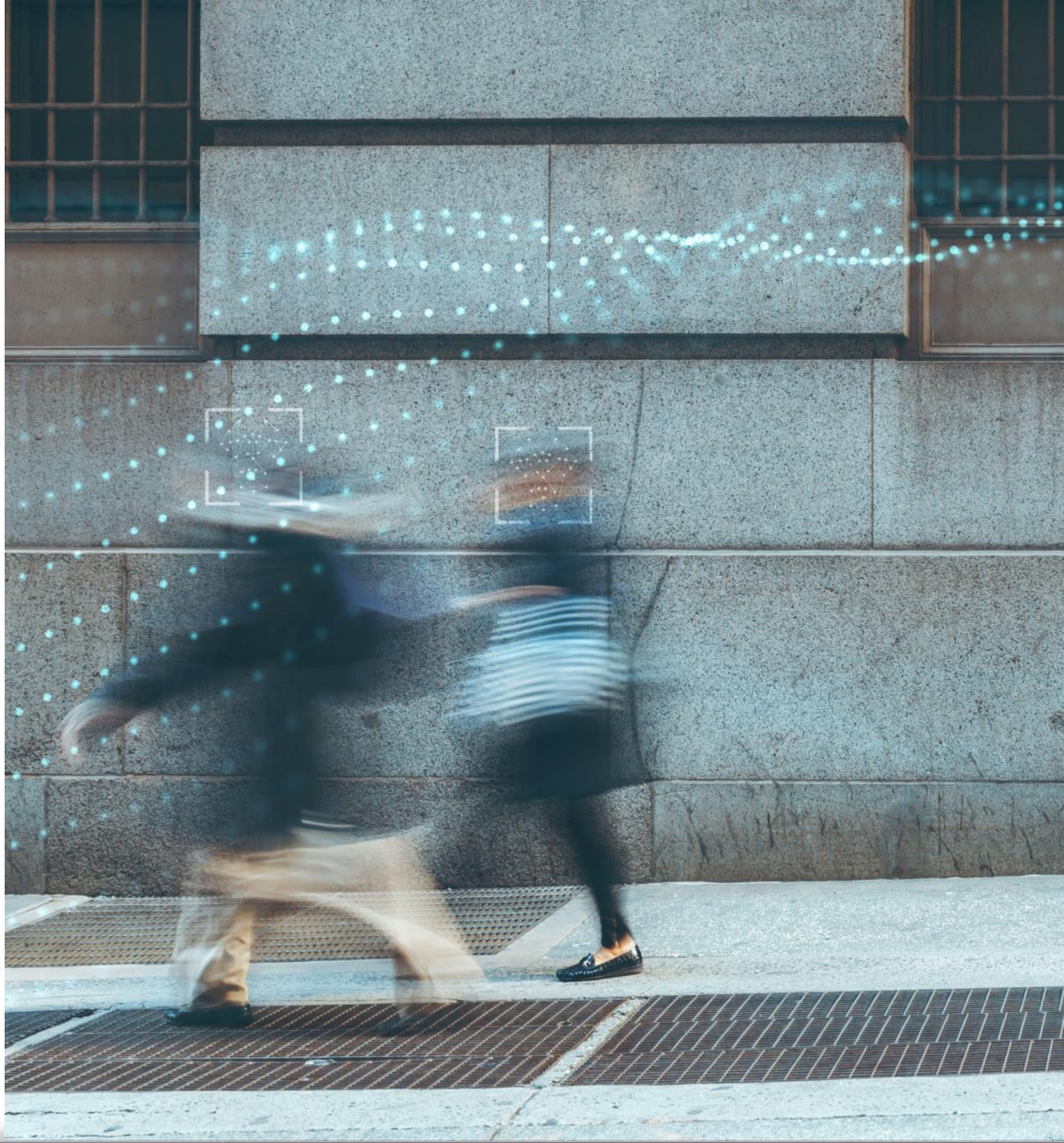
- Checkboxes are used for toggling data values (true/false)
- General format is *<input type="checkbox" v-model="<model prop>" />*
- Model property often a Boolean value
- Alternatively, can use *true-value* and *false-value* for different values

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

```
<input type="checkbox" v-model="benefitsSelected" true-value="yes" false-value="no"> Benefits
```

v-model with Drop-down Lists

- In HTML, drop-down lists include a <*select* /> and <*option* /> elements
- The <*select* /> element stores selected value
- Use *v-model* to bind model property to <*select* /> element for dynamic updates



v-model with Drop-down Lists

- Can use a collection and *v-for* to dynamically build options
- Use *:value* to associate an "internal", unique value to the selected option
- Bind contents of option to string value for display
- Can use either index/array contents or properties if collection of objects

```
<select v-model="statusIndex">
    <!-- Create a message to select one -->
    <option disabled value="">Please select one</option>
    <!-- Use v-for to create the list of options -->
    <option v-for="(status, index) in statusList" :value="index">
        {{ status }}
    </option>
</select>
```

```
<select v-model="selectedValue">
   <option v-for="item in items" :value="item.value">
   {{ item.displayProperty }}
   </option>
</select>
```

Reactor

Working with Events

Working with Events

- Events can be used to respond programmatically to user actions
- Examples include clicking a button, selecting a value from a drop-down, etc.
- Code event handlers to encapsulate logic that should occur in response

v-on (or the @ Shortcut)

- Bind an element's event by name using the *v-on* directive
- @ can be used as a shortcut for *v-on*
- Handler will be a JavaScript function

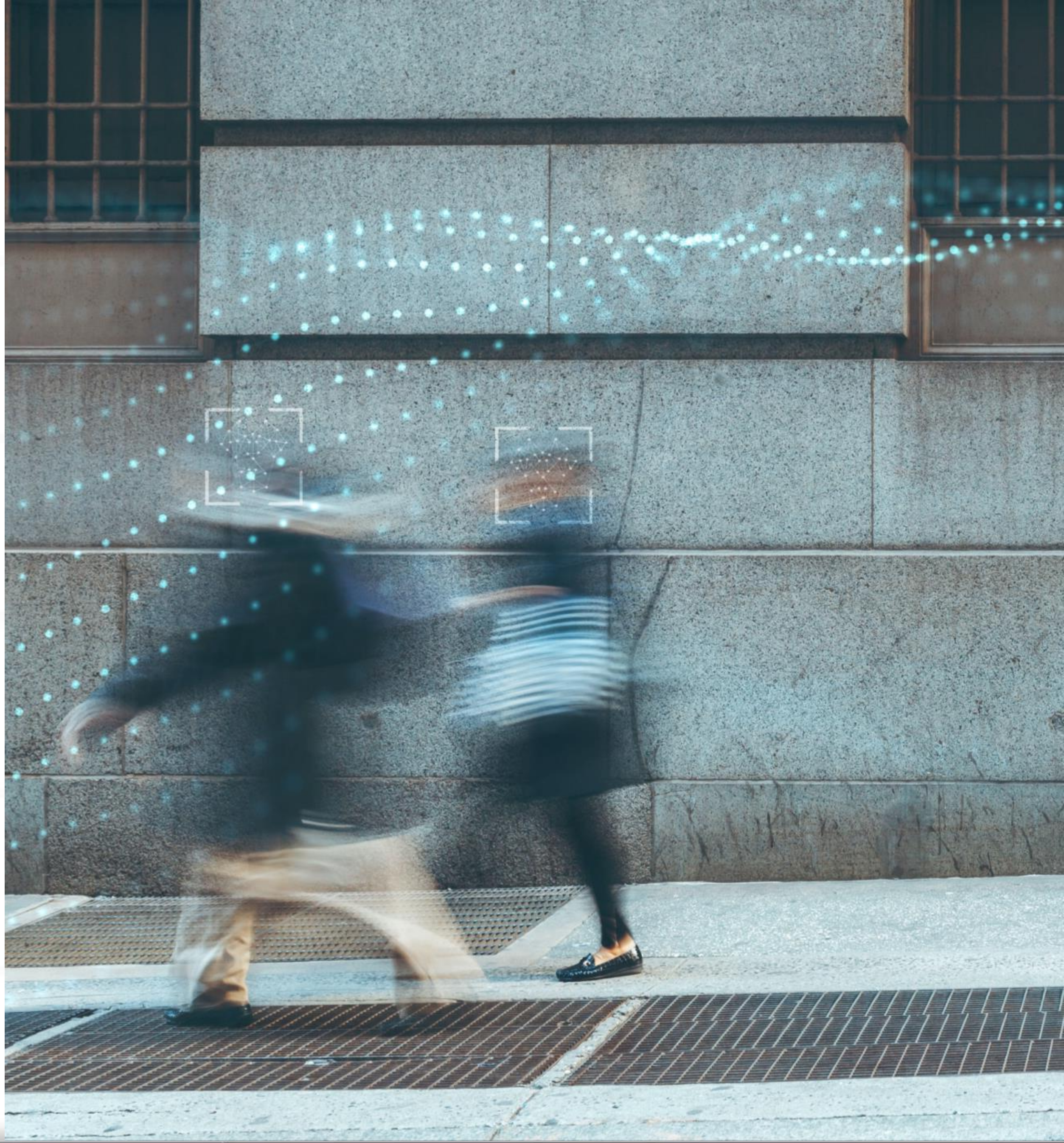
methods Property

- In Vue, handlers are defined in *methods* property of app
- Can be called by name like any other JavaScript function
- Handlers defined in *methods* able to access any registered data in the model

```
const app = Vue.createApp({
    data() {
        return {
            name: 'Cheryl'
        }
    },

    methods: {
        displayName() {
            console.log(this.name);
        }
    }
});
```

```
<button type="button" @click="displayName">Display name</button>
```



Working with Computed Properties

Microsoft Reactor

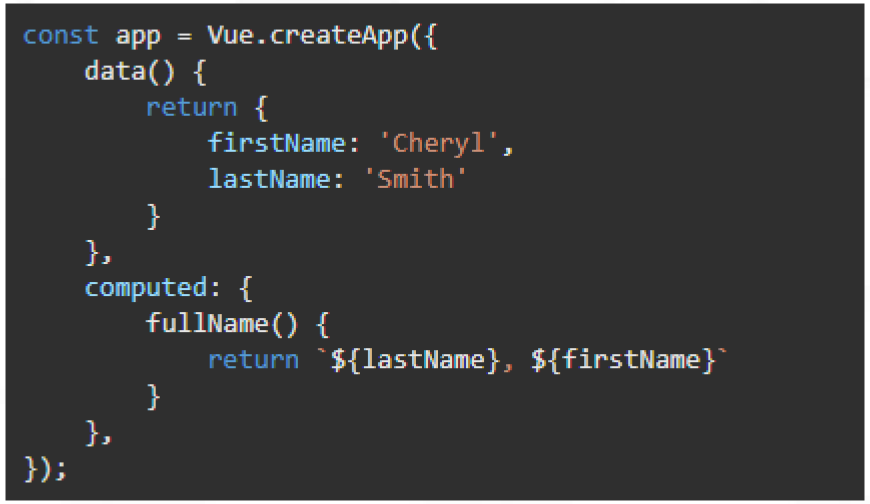
Computed Properties

- Functions that return a value
- Defined in *computed* property of Vue app
- Enables more sophisticated build-out of model data for binding

Microsoft Reactor

Computed Properties

- As a function, can do more than just simple data property reference (logic)
- If model properties used to compute are changed, computed property updates
- What is known as “reactive”



```
const app = Vue.createApp({
    data() {
        return {
            firstName: 'Cheryl',
            lastName: 'Smith'
        }
    },
    computed: {
        fullName() {
            return `${lastName}, ${firstName}`
        }
    },
});
```

Knowledge Check

Summary

Microsoft Reactor

What Did We Cover?

- Dynamic rendering of lists in Vue and conditional rendering
- Working with forms, two-way binding, and events
- Using computed properties in our Vue.js app

Next Steps / Further Study

- https://aka.ms/VueDynamicRendering
- https://aka.ms/VueDataEvents
- https://vuejs.org/
- https://v3.vuejs.org/guide/introduction.html
- https://www.vuemastery.com/blog/vue-2-or-vue-3/

Microsoft

Reactor

Thank You!

Q&A

https://www.meetup.com/pro/microsoft-reactor

@MSFTReactor

http://www.youtube.com/c/MicrosoftReactor

aka.ms/ReactorEmailSignUp

Microsoft

Reactor

We are constantly striving to create excellent content and would appreciate if you could take this brief survey.

Survey Link: https://aka.ms/Reactor/Survey

Please enter the event code 13101 at the start of survey

Microsoft