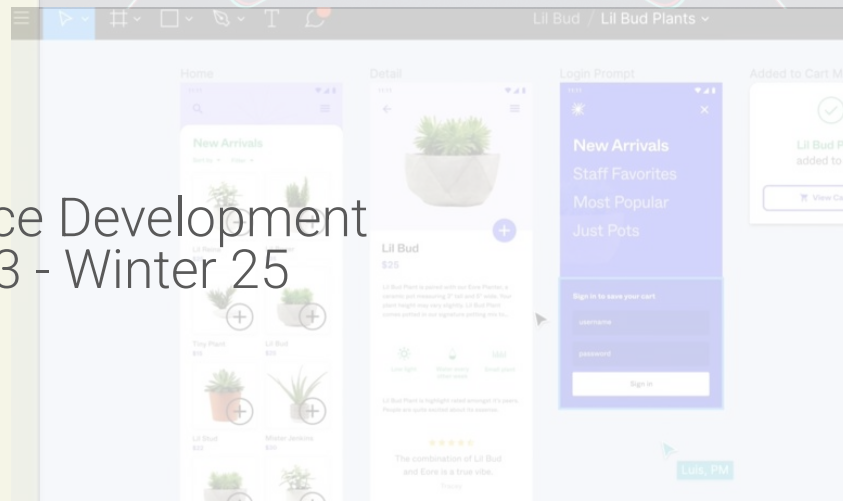





The DESIGN
of EVERYDAY
THINGS

Vue.js Framework

User Interface Development
EECS 493 - Winter 25



Class progress

1. User research 
 - a. Assignment 1, Final Project all milestones
2. Web programming 
 - a. Assignment 2, 3, 5 (JavaScript, jQuery)
3. Design and prototyping 
 - a. Assignment 4, Final Project milestones 3-4

Class progress



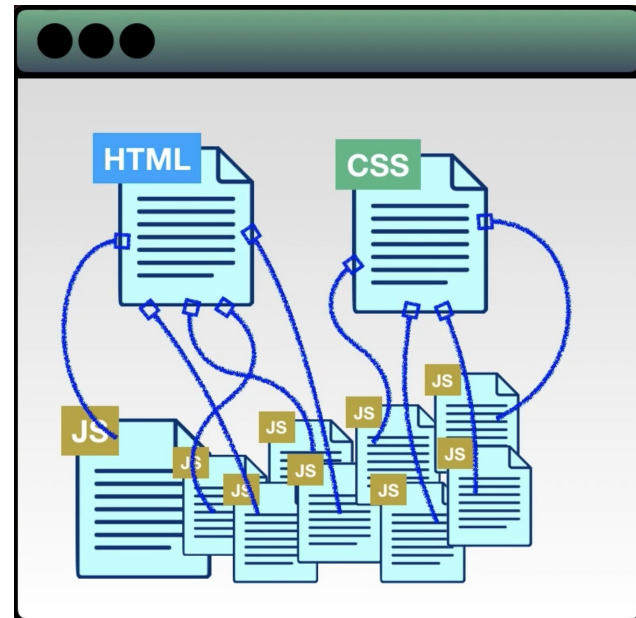
1. User research
 - a. Assignment 1, Final Project all milestones
2. **Web programming**
 - a. Assignment 2, 3 (JavaScript, jQuery)
 - b. Assignment 5 (Vue.js framework)**
3. Design and prototyping
 - a. Assignment 4, Final Project milestones 3-4

Goals for today:

1. The benefit of using Vue compared to vanilla JavaScript
2. How to create a Vue instance
3. Attribute binding in Vue
4. Conditional rendering, event handling in Vue

What is Vue

A JavaScript Framework for building more approachable, versatile, performant web pages



Same code in plain JavaScript vs Vue.js

plain JavaScript

Use element selector to update component

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4  <title>Product App</title>
5  </head>
6
7  <body>
8    <div id = "app">
9      <h1>My product:</h1>
10     <h1 id = "product-info"></h1>
11   </div>
12 </body>
13
14 <script>
15   document.getElementById("product-info").
16     innerHTML = "Boots";
17 </script>
18 </html>
```

Same code in plain JavaScript vs Vue.js

Vue.js

Create a Vue application →

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>Product App</title>
5 <script src="https://unpkg.com/vue@3"></script>
6 </head>
7
8 <body>
9   <div id = "app">
10     <h1>My Product:</h1>
11     <h1>{{ product }}</h1>
12   </div>
13 </body>
14
15 <script>
16   const { createApp } = Vue;
17
18   createApp({
19     data(){
20       return {
21         product: 'Boots',
22       }
23     },
24   }).mount('#app');
25 </script>
26 </html>
```

Same code in plain JavaScript vs Vue.js

Vue.js

Create a Vue application →

Defines the data object for the Vue instance. It returns an object that contains the "product" property →

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>Product App</title>
5 <script src="https://unpkg.com/vue@3"></script>
6 </head>
7
8 <body>
9   <div id = "app">
10     <h1>My Product:</h1>
11     <h1>{{ product }}</h1>
12   </div>
13 </body>
14
15 <script>
16   const { createApp } = Vue;
17
18   createApp({
19     data(){
20       return {
21         product: 'Boots',
22       }
23     },
24   }).mount('#app');
25 </script>
26 </html>
```

Same code in plain JavaScript vs Vue.js

Vue.js

- Create a Vue application →
- Defines the data object for the Vue application. It returns an object that contains the "product" property →
- Mount the Vue application to DOM →

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>Product App</title>
5 <script src="https://unpkg.com/vue@3"></script>
6 </head>
7
8 <body>
9   <div id = "app">
10     <h1>My Product:</h1>
11     <h1>{{ product }}</h1>
12   </div>
13 </body>
14
15 <script>
16   const { createApp } = Vue;
17
18   createApp({
19     data() {
20       return {
21         product: 'Boots',
22       }
23     },
24     }.mount('#app');
25 </script>
26 </html>
```

Same code in plain JavaScript vs Vue.js

Vue.js

- Bind the view with the data, it'll display the value of the 'product' data property in the Vue instance →
- Create a Vue application →
- Defines the data object for the Vue application. It returns an object that contains the "product" property →
- Mount the Vue application to DOM →

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>Product App</title>
5 <script src="https://unpkg.com/vue@3"></script>
6 </head>
7
8 <body>
9   <div id = "app">
10     <h1>My Product:</h1>
11     <h1>{{ product }}</h1>
12   </div>
13 </body>
14
15 <script>
16   const { createApp } = Vue;
17
18   createApp({
19     data() {
20       return {
21         product: 'Boots',
22       }
23     },
24     }.mount('#app');
25 </script>
26 </html>
```


Same code in plain JavaScript vs Vue.js

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>Product App</title>
5 </head>
6
7 <body>
8   <div id = "app">
9     <h1>My product:</h1>
10    <h1 id = "product-info"></h1>
11  </div>
12 </body>
13
14 <script>
15 document.getElementById("product-info").
  innerHTML = "Boots";
16 </script>
17 </html>
```

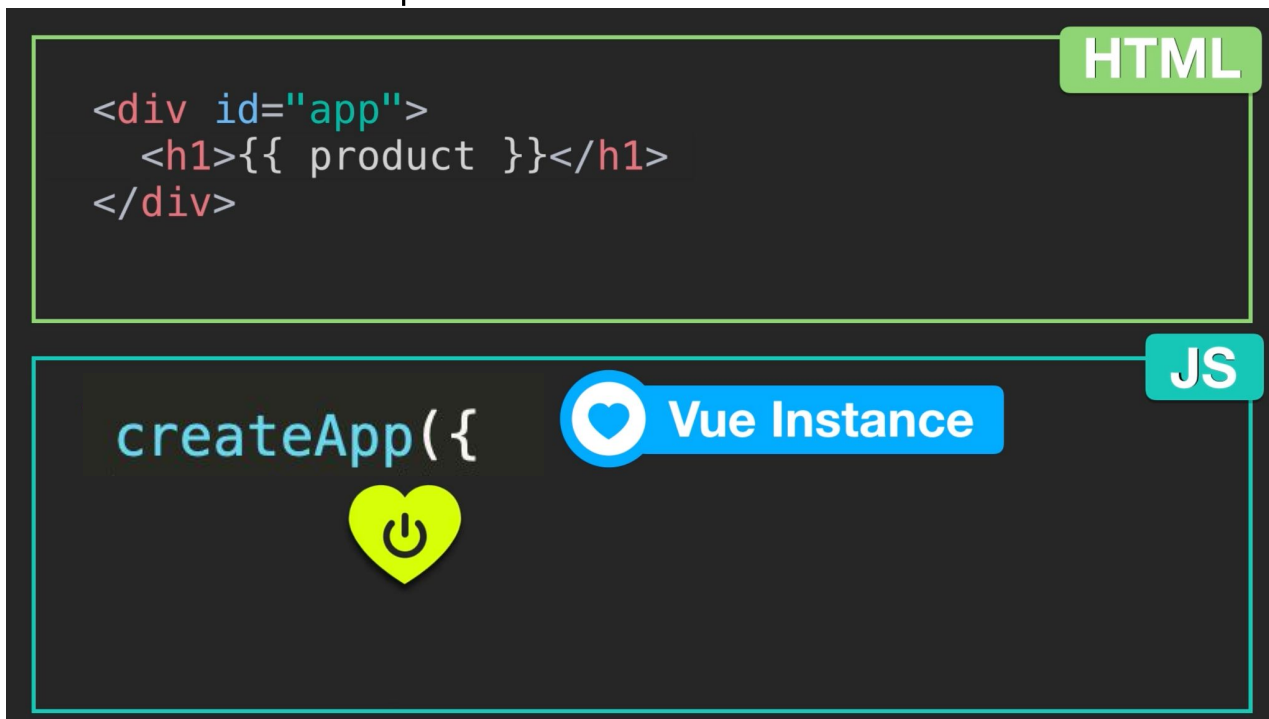
Need to do
document.getElementById() every
time I want to update the data

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>Product App</title>
5 <script src="https://unpkg.com/vue@3"></script>
6 </head>
7
8 <body>
9   <div id = "app">
10    <h1>My Product:</h1>
11    <h1>{{ product }}</h1>
12  </div>
13 </body>
14
15 <script>
16   const { createApp } = Vue;
17
18   createApp({
19     data(){
20       return {
21         product: 'Boots',
22       }
23     },
24   }).mount('#app');
25 </script>
26 </html>
```

Bind view with data, so it's automatically updated

Creating a Vue Instance

The Vue instance powers the connection between data and view



Connect the Vue instance to an HTML element

The Vue instance is responsible for passing data into the HTML element



Use expressions in HTML to receive data

This diagram shows how to use expressions in HTML to receive data from a JavaScript instance. It features a **JS** label in the top right corner.

HTML Expression:

```
<h1>{{ product }}</h1>
```

Expression Definition:

{{ }} **Expression**

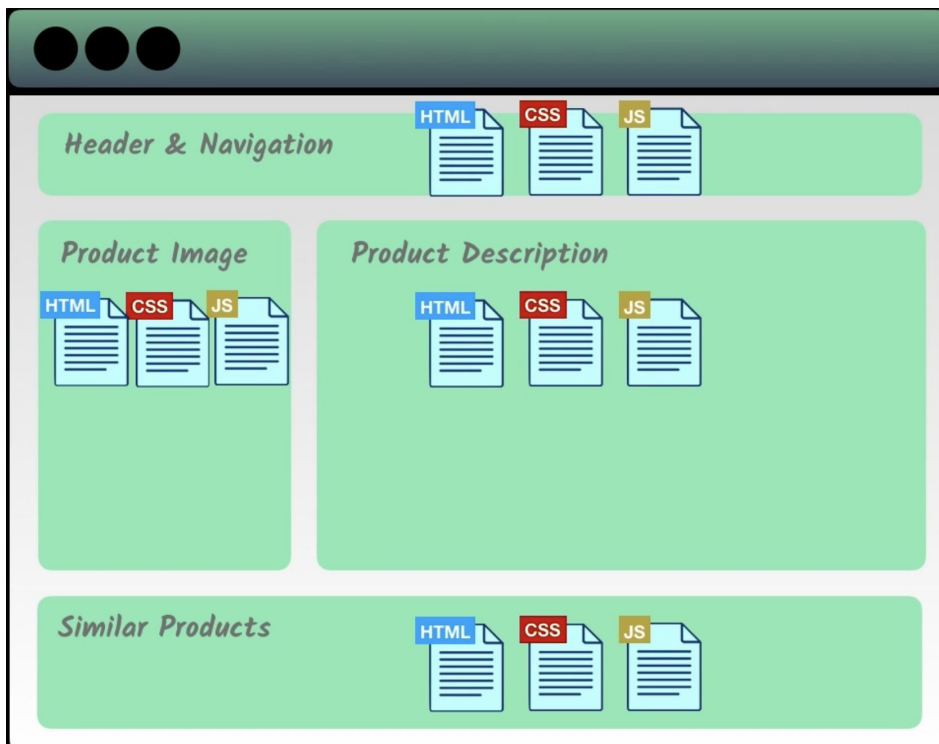
Examples of valid expressions:

- `{{ product + '?' }}`
- `{{ firstName + ' ' + lastName }}`
- `{{ clicked ? true : false }}`
- `{{ message.split('').reverse().join('') }}`

Livcoding example

index-vue.html

Vue splits a webpage into reusable components.



Vue.js - Directives

Directives help coordinate program flow

We will discuss

- v-bind
- v-if
- v-for
- v-on

Directives can be added to any DOM elements

Attribute Binding
v-bind



Bind attribute to an HTML element



My product: Tshirt

Bind attribute to an HTML element



My product: Tshirt

```
10 <body>
11   <div id = "app">
12     <div class = "product">
13       <div class = "product-image">
14         
15       </div>
16       <div class = "product-info">
17         <h1>My product: {{product}}</h1>
18       </div>
19     </div>
20   </div>
21 </body>
22
23 <script>
24   const { createApp } = Vue;
25
26   createApp({
27     data(){
28       return {
29         product: 'Tshirt',
30         image: './assets/shirt-yellow.jpeg'
31       }
32     },
33   }).mount('#app');
34 </script>
```

A variety of attributes

Syntax:

v-bind: or :



Examples

```
:alt="description"  
:href="url"  
:title="toolTip"  
:class="isActive"  
:style="isStyled"  
:disabled="isDisabled"
```

Livecoding example

attribute-binding-url.html

Lecture 16-Survey 1

<https://forms.gle/pzg7t47BGrnoHLUo9>

www.yellkey.com/alone



Which are correct about Vue.js (Select all that apply) *

- ☐ Vue.js simplifies the development of web applications through binding data and view
 - ☐ Vue.js is not able to make changes to the DOM of a web page
 - ☐ One webpage can only have one Vue instance, so that it creates the 1-1 mapping between data and view
 - ☐ Vue.js supports the Virtual DOM, allowing for efficient updates and rendering of the webpage
 - ☐ Vue.js requires a full page reload for updating the view whenever the data changes.
-

Which of the following is NOT a correct syntax in Vue *

- ☐ :src="image"
 - ☐ v-bind:src={{image}}
 - ☐ {{text}}
 - ☐ v-bind:style="{color:textColor}"
-

Use of curly braces

- {{product}} -> display the value of this data property
- v-bind:style = "{color:textColor}" -> in directive, {} defines an object, it's more than a string
- v-bind:href = "url" -> in directive, when the data is only a string, no curly braces

Conditional Rendering

v-if, v-show



Add conditionals for rendering



Tshirt

Out of Stock

Add conditionals for rendering



Tshirt

Out of Stock

```
<body>
  <div id = "app">
    <div class = "product">
      <div class = "product-image">
        
      </div>
      <div class = "product-info">
        <h1>{{product}}</h1>
        <p v-if="inventory>10">In Stock</p>
        <p v-else-if="inventory<=10 && inventory
          >0">Almost sold out</p>
        <p v-else>Out of Stock</p>
      </div>
    </div>
  </div>
</body>
```

Livcoding example

conditional.html

Difference between v-if and v-show

- v-if inserts or deletes the item
- v-show doesn't remove DOM elements, it changes the visibility of elements

Lecture 16-Survey 2

<https://forms.gle/hiJiGRdKJqoiRVgE6>

www.yellkey.com/attorney



Consider the following code. What will be displayed in the browser? *

```
<body>
  <div id = "app">
    <p v-if="showMessage">Hello, Vue!</p>
  </div>
</body>

<script>
  const { createApp } = Vue;
  createApp({
    data(){
      return {
        showMessage: false,
      }
    },
  }).mount('#app');
</script>
```

- ☐ Hello, Vue!
- ☐ Nothing
- ☐ <p v-if="false">Hello, Vue!</p>
- ☐ <p v-if="showMessage">Hello, Vue!</p>

List Rendering v-for



PalmPilot wooden model

Render a list



Tshirt

Almost sold out

- 80% cotton
- 20% polyester
- Gender-neutral

```
<ul>
  <li v-for="detail in details">
    {{detail}}</li>
</ul>
```

```
<script>
const { createApp } = Vue;
createApp({
  data(){
    return {
      product:'Tshirt',
      image:"./assets/shirt-yellow.jpeg",
      inventory:10,
      details:["80% cotton", "20% polyester", "
        Gender-neutral",],
    }
  },
}).mount('#app');
```

Render a list of objects

listrendering.html



Tshirt

Almost sold out

- 80% cotton
- 20% polyester
- Gender-neutral

Item in stock:

- Color: blue
- Size: XL

Item in stock:

- Color: yellow
- Size: S

Render a list of objects

```
<div v-for="(item,index) in items" :id="item.itemID">
  <p>Item {{index+1}} in stock:
    <ul>
      <li>Color: {{item.itemColor}}</li>
      <li>Size: {{item.itemSize}}</li>
    </ul>
  </p>
</div>
```

```
items:[
  {
    itemID: 2234,
    itemColor: "blue",
    itemSize: "XL"
  },
  {
    itemID: 2235,
    itemColor: "yellow",
    itemSize: "S"
  },
]
```

- Use dot notation to reference properties in each object
- Give each node an identity (:id)
- Use index to indicate the position in the array (it can be a different name)

Lecture 16-Survey 3

<https://forms.gle/yBUDG6XKAbrDomcH9>

www.yellkey.com/agree



Consider the following code. What will be displayed in the browser? *

```
<body>
  <div id = "app">
    <li v-for="(item, idx) in items">
      Item {{idx+1}}: {{item.name}}</li>
    </div>
  </body>

<script>
  const { createApp } = Vue;
  createApp({
    data(){
      return {
        items:[{name:'Apple', color:'red'},
               {name:'Banana', color:'yellow'},
               {name:'Cherry', color:'red'},
               ]
      }
    },
  }).mount('#app');
</script>
```

- Item 1: Apple
- Item 2: Banana
- Item 3: Cherry

☐ Option 1

- Item : Apple
- Item : Banana
- Item : Cherry

☐ Option 2

- Item : Apple, red
- Item : Banana, yellow
- Item : Cherry, red

☐ Option 3

- name : Apple, color: red
- name : Banana, color: yellow
- name : Cherry, color: red

☐ Option 4

Event Handling

v-on, @



PalmPilot wooden model

Add event listener to button



Tshirt

Almost sold out

- 80% cotton
- 20% polyester
- Gender-neutral

Add to Cart

Cart:2

Add event listener to button

```
<button v-on:click="addToCart">Add to Cart</button>
<div class="cart">
  <p>Cart:{{cart}}</p>
</div>
```

```
createApp({
  data() {
    return {
      product: 'Tshirt',
      image: './assets/shirt-yellow.jpeg',
      inventory: 10,
      cart: 0,
    }
  },

```

```
methods: {
  addToCart: function() {
    this.cart += 1
  },
}
```

Which are correct about Vue.js (Select all that apply) *

- ☐ Vue.js simplifies the development of web applications through binding data and view
 - ☐ Vue.js is not able to make changes to the DOM of a web page
 - ☐ One webpage can only have one Vue instance, so that it creates the 1-1 mapping between data and view
 - ☐ Vue.js supports the Virtual DOM, allowing for efficient updates and rendering of the webpage
 - ☐ Vue.js requires a full page reload for updating the view whenever the data changes.
-

Livcoding example

eventhandling.html

Structure in the Vue application

```
const { createApp } = Vue;
createApp({
  data(){
    return {
      product: 'Tshirt',
      image: './assets/shirt-yellow.jpeg',
      inventory: 10,
      cart: 0,
    }
  },
  methods: {
    addToCart: function(){
      this.cart += 1
    },
  },
}).mount('#app');
```

Data defines all the data properties the web page needs

Method defines all functions

More on event listeners

- v-on can be shortened as @
- v-on:click is the same as @click
- Methods take parameters

```
<div v-for="item in items" :key="item.itemID">
  <p @mouseover="updateProduct(item.itemImage)">
    {{item.itemColor}}</p>
</div>
```

```
updateProduct: function(itemImage){
  this.image = itemImage
}
```

Lecture 16-Survey 4

<https://forms.gle/NEjjLtMoQUmfVATH7>

www.yellkey.com/article



```
<body>
  <div id = "app">
    <button [put your code here]>Display Welcome
    Message!</button>
  </div>
</body>

<script>
  const { createApp } = Vue;
  createApp({
    data(){
      return {
        personName: 'Alice',
      }
    },
    methods:{
      greet: function(personName){
        alert("Welcome "+ personName);
      }
    }
  }).mount('#app');
</script>
```

- ☐ v-on:click = "{personName}"
- ☐ @click="greet(personName)"
- ☐ @click="Alice"
- ☐ v-on:click="greet()"

Goals for today:

1. The benefit of using Vue compared to vanilla JavaScript
2. How to create a Vue instance
3. Attribute binding in Vue
4. Conditional rendering, event handling in Vue

