

0 WHY VUE ?

1 DATA BINDING

2 METHODS & EVENTS

3 LOOPS / IF/ELSE

4 COMPUTED & WATCHERS

5 CLI

6 COMPONENTS

7 ROUTING

8 SLOTS / WIDGETS

9 AUTHENTICATION

X VUTIFY



ROADMAP



VUE

DYNAMIC STYLE

&

COMPUTED



What will we learn ?



- ✓ How to render style **dynamically**
- ✓ How to use **methods** or **computed** data

:class

computed :{}

methods :{}



10 MIN



CLASS

1-demo- class binding

Change the style of the div
To active is div is selected

Vue Dynamic Styling

Vue Dynamic Styling

A

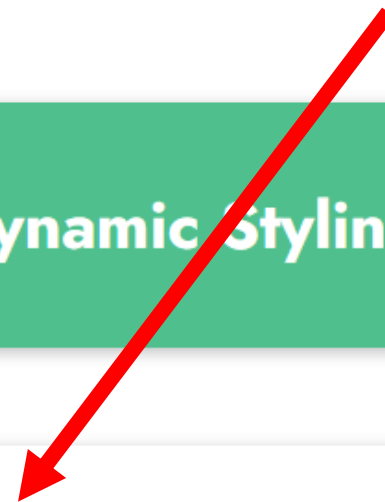
CLICK



CLICK



A





10 MIN



CLASS

V-BIND ON CLASS

If `isValid` is true, the `green` class will be applied



```
<h1 :class="{ green: isValid }">...</h1>
```

```
data() {  
  return {  
    isValid : true,  
  }  
}
```



10 MIN

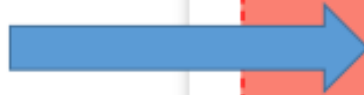


INDIV

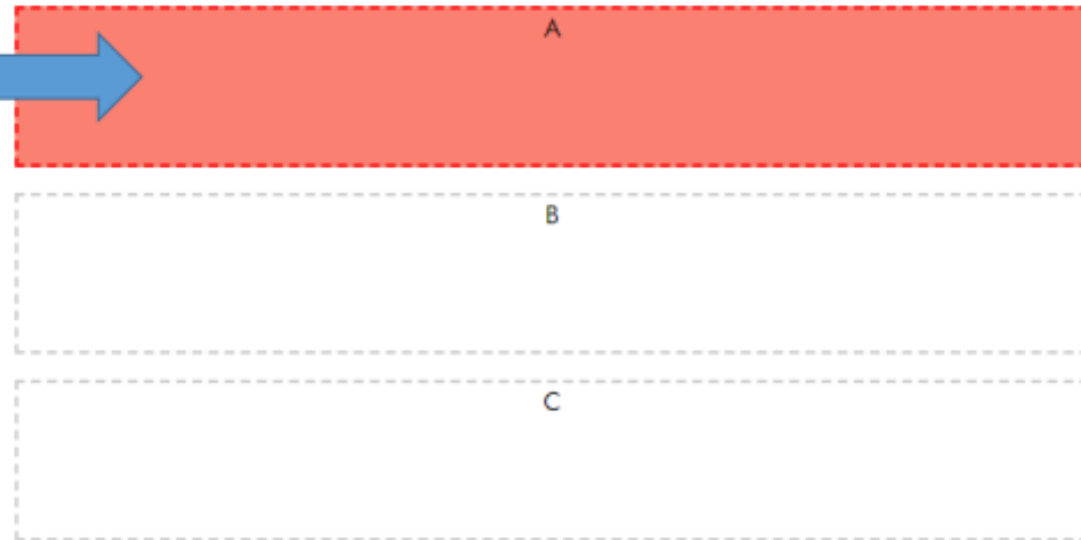
Activity 1

Vue Dynamic Styling

A is RED if Selected



Do the same of B and C





What will be the resulted view ?

```
data() {  
  return {  
    isValid: false,  
    isFinished: true,  
    isCrazy: true,  
  };  
}
```

```
<div  
  class="card"  
  :class="{  
    green: isValid,  
    flag: isFinished,  
    star: isCrazy,  
  }"  
  
></div>
```

- A** <div class="card green flag star">
- B** <div class="green flag star">
- C** <div class="card flag star">
- D** <div class="flag star">



What will be the result ?

```
data() {  
  return {  
    isValid: false,  
    isFinished: true,  
    isCrazy: true,  
  };  
}
```

```
<div  
  class="card"  
  :class="{  
    green: isValid,  
    flag: isFinished,  
    star: isCrazy,  
  }"  
></div>
```

permanent

dynamic

A <div class="card green flag star">

B <div class="green flag star">

C <div class="card flag star">

we combine dynamic and permanent styles

D <div class="flag star">



What will be the result ?

```
data() {  
  return {  
    isEmpty: false,  
    isCrazy: true,  
  };  
}
```

```
<div  
  class="box hover"  
  :class="{  
    warning: isEmpty,  
    alert: isCrazy  
  }"  
></div>
```

- A** <div class="box hover alert">
- B** <div class="alert">
- C** <div class="box hover warning">
- D** <div class="box hover isCrazy">



What will be the result ?

```
data() {  
  return {  
    isEmpty: false,  
    isCrazy: true,  
  };  
}
```

```
<div  
  class="box hover"  
  :class="{  
    warning: isEmpty,  
    alert: isCrazy  
  }"  
  
></div>
```

- A** <div class="box hover alert">
- B** <div class="alert">
- C** <div class="box hover warning">
- D** <div class="box hover isCrazy">



10 MIN



CLASS

3-demo- method and computed

Vue Computed

Your Name: ronan Ogor

When text filed changes, we
update the full name

This work but, let's move the logic on the JS !!

ACTIVITY 2

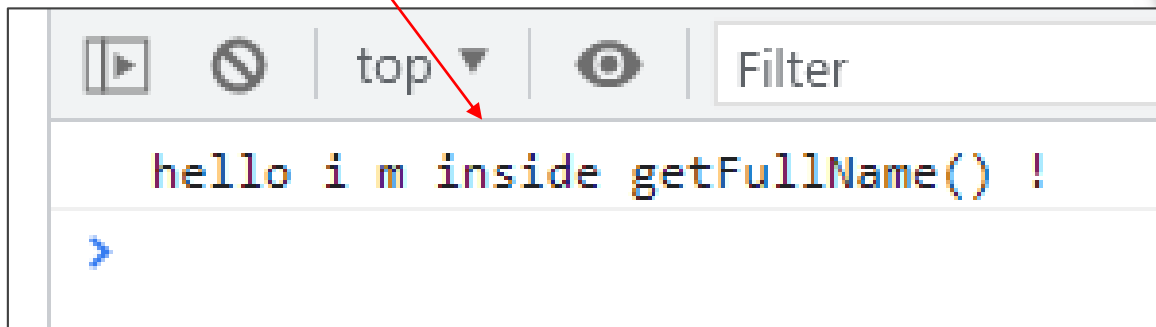


05 MIN



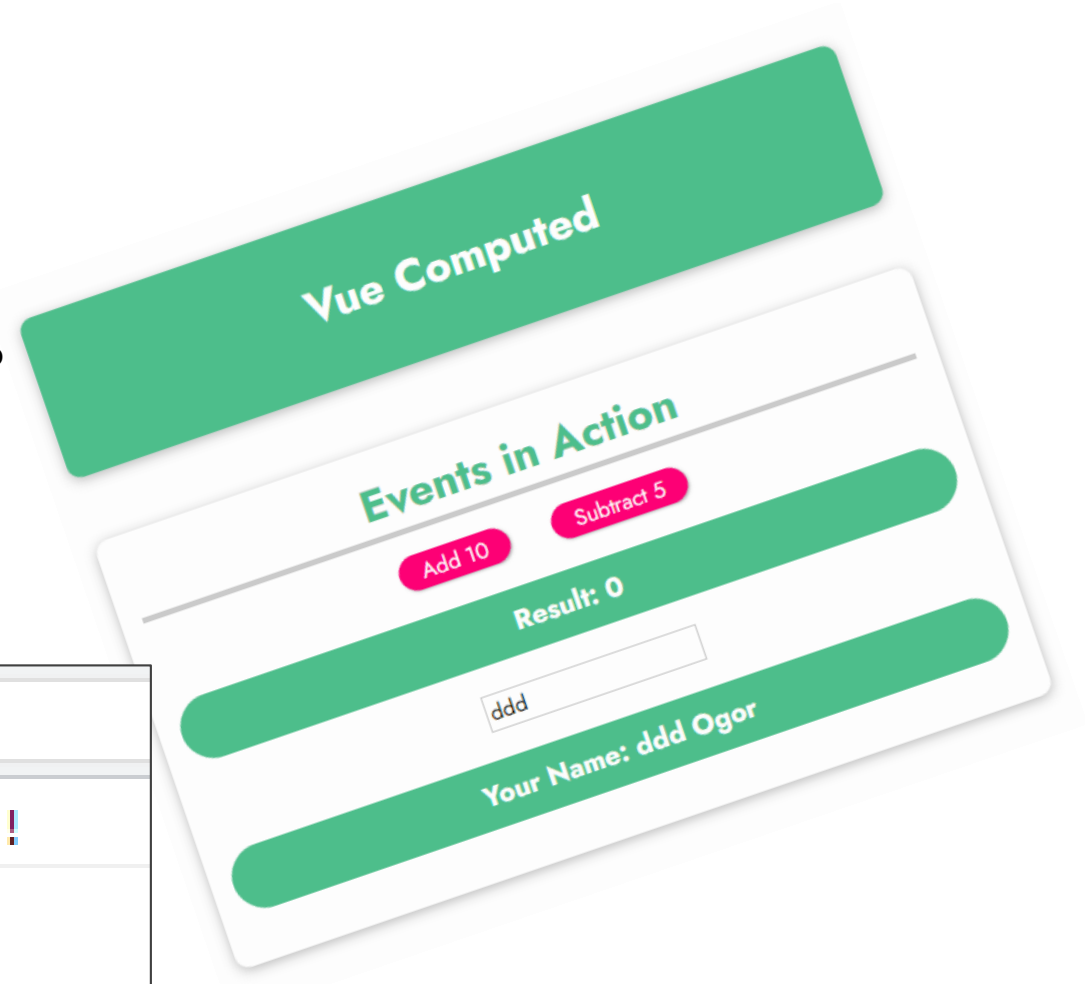
INDIV

- ✓ Do you call `getFullName()` when you refresh the page ?
- ✓ Do you call `getFullName()` when you click on Add 10 ?
- ✓ Do you call `getFullName()` when you click on Subtract 5 ?
- ✓ Do you call `getFullName()` when you change the TextField ?



The screenshot shows a web browser's developer console. The top bar contains icons for opening the console, disabling it, and a dropdown menu currently set to 'top'. To the right of the dropdown is a 'Filter' input field. The console area displays a single log entry: 'hello i m inside getFullName() !' in a monospaced font. A red arrow points from the text 'when you change the TextField ?' in the list above to the 'top' dropdown menu in the console toolbar.

```
hello i m inside getFullName() !
```





05 MIN



INDIV

ACTIVITY 3

Define `getFullName()` as a **computed data** instead of a method

```
computed: {  
  getFullname() {  
  },  
},
```

- ✓ Do you call `getFullName()` when you refresh the page ?
- ✓ Do you call `getFullName()` when you click on Add 10 ?
- ✓ Do you call `getFullName()` when you click on Subtract 5 ?
- ✓ Do you call `getFullName()` when you change the TextField ?



10 MIN



CLASS

methods VS computed!

```
methods: {  
  c() {  
    return this.a + this.b;  
  },  
},
```



*Called whenever something
Change on page*

```
computed: {  
  c() {  
    return this.a + this.b;  
  },  
},
```



Called only if a or b change



methods

Functions that are meant to be called

Can have parameters passed in

Re-evaluated every time they're called

Not meant to be a property

computed

Intended to compute existing data

Shouldn't have parameters passed in

Cached based on their dependencies

Should be used as a property



Dynamic class & computed

We can also bind a style to a **computed property** that returns an **object**.

```
<div :class="divStyle"></div>
```

```
data() {  
  return {  
    isActive: true,  
    value: 0  
  }  
},  
computed: {  
  divStyle() {  
    return {  
      active: this.isActive,  
      danger: this.value < 5  
    }  
  }  
}  
}
```




What will be the result ?

```
<div  
  class ="demo"  
  :class="divStyle"  
></div>
```

```
data() {  
  return {  
    name: "ronan",  
    age: 19,  
    selected : true  
  }  
},  
computed: {  
  divStyle() {  
    return {  
      full: this.selected,  
      warning: this.age < 18,  
      great : this.name === "ronan"  
    }  
  }  
}  
}
```

- A** <div class= "demo full warning">
- B** <div class= "full warning great">
- C** <div class= "demo full great">
- D** <div class= "full great">



What we have learnt ?



- ✓ How to render style **dynamically**
- ✓ How to use **methods** or **computed** data

:class

computed :{}

methods :{}