Moving React to Vue Hooks

ARE WE GETTING HOOKED? 🌸

bit.ly/jsmd-hooks-to-capi

🌸 Andrew Luca

❤️ All In Developer & Open sourcerer

👨‍💻 Tech Lead Roata Wăy, Code for Moldova, Jagaad

📗 Coach Jagaad Academy

Find me on **iamandrewluca.com**

What are XYZ hooks? 🪝

Hooks are functions that let you “hook into” XYZ state and lifecycle features from function components.

- React 16.8 (hooks)
- Vue 3 (Composition API)
- Svelte (no name)
- Solid (same as React)
- Angular 14?! (DI and inject)

```
let { lorem, ipsum, dolor } = useCustomHook()
```

JSMD12 Vue composition API
bit.ly/jsmd-capi

Comparison with React Hooks 👌

React 💙

- Repeated invoking
- Order and conditionals
- Closure, stale, dependencies
- Memoized callbacks, useEvent
- Stale, concurrent, rendering

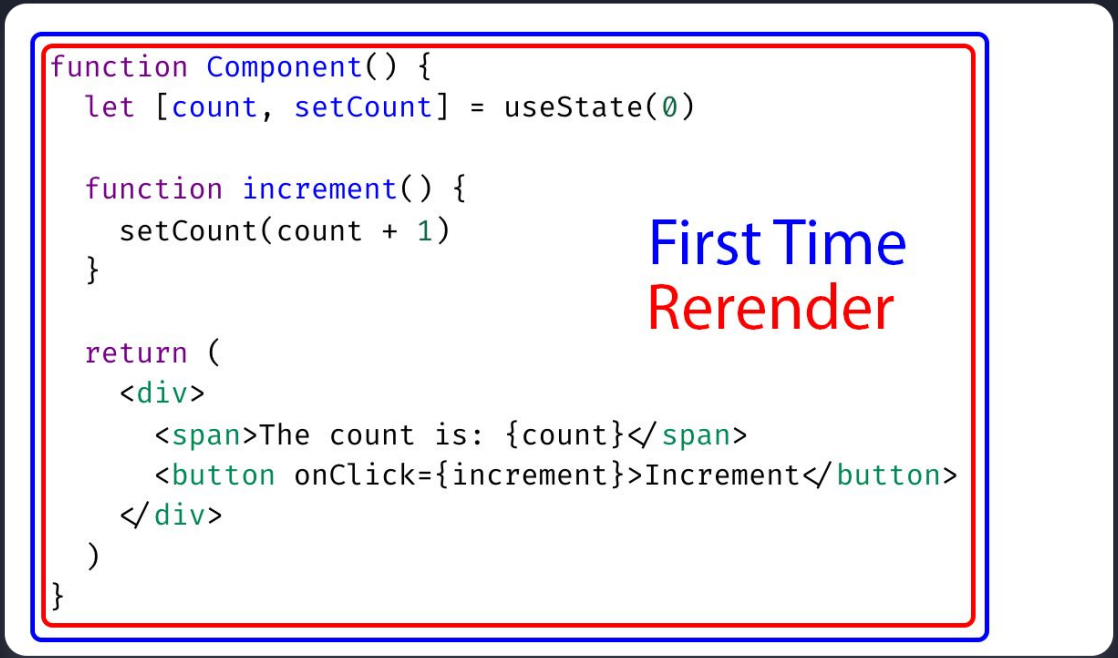
Vue 💚

- Invokes only once
- Reactivity, no dependencies
- No cached callbacks

Comparison with React Hooks

bit.ly/vue-hooks-vs-capi

What is actually rendering in React? ⚙️



What is actually rendering in Vue? ⚙️

⚠️ no SFC

```
let Component = defineComponent({
  setup() {
    let count = ref(0)

    function increment() {
      count.value = count.value + 1
    }

    return () => (
      <div>
        <span>The count is: {count}</span>
        <button onClick={increment}>Increment</button>
      </div>
    )
  }
})
```

First Time
Rerender

Are they really Hooks? 👀

React 💙

- **useState**
- **useRef**
- **useMemo**
- **useCallback**
- **useEffect**
- **useLayoutEffect**
- **useContext**
- **useReducer**

Vue 💚

- **reactive / ref**
- **computed**
- **watchEffect**
- **provide / inject**
- **pinia.vuejs.org**
- **onMounted**
- **onUnmounted**
- **onUpdated**
- ... much more

They are hooks in disguise! 🕵🏼

React 💙

- useState
- useRef
- useMemo
- useCallback
- useEffect
- useLayoutEffect
- useContext
- useReducer

Vue 💚

- reactive / ref
- computed
- watchEffect
- provide / inject
- pinia.vuejs.org
- onMounted
- onUnmounted
- onUpdated
- ... much more

What we will look over exactly? 🤔

- State (useState, useReducer, ref, reactive, Pinia)
- Context (useContext, provide, inject)
- Memoization (useMemo, useCallback, computed)
- Effects (useEffect, useLayoutEffect, watchEffect)
- Lifecycle (useEffect, useLayoutEffect, onMounted, onUnmounted, ...)
- References (useRef, ref)

State / React

```
let [state, setState] = useState(0)

// somewhere later
setState(1)
```

```
let reducer = (state, action) ⇒ state + action
let [state, dispatch] = useReducer(reducer, 0)

// somewhere later
dispatch(1)
```

State / Vue

```
let state = ref(0)

// somewhere later
state.value = state.value + 1
```

```
let state = reactive({ count: 0 })

// somewhere later
state.count = state.count + 1
```

⚠️ problems with native objects

```
let useCounterStore = defineStore('counter', {
  state: () => ({ count: 0 }),
  actions: {
    increment() {
      this.count++
    },
  },
})

// somewhere later
let counter = useCounterStore()
counter.increment()
```

Context / React

```
let Context = createContext()

function Providers({ children }) {
  return (
    <Context.Provider value={0}>
      {chidlren}
    </Context.Provider>
  )
}

// somewhere later
let value = useContext(Context)
```

Context / Vue

```
let Context = Symbol("count")

let App = defineComponent({
  setup() {
    provide(Context, 0)
  }
})

// somewhere later
let value = inject(Context)
```

Memoization / React

```
let [count, setCount] = useState(0)

let doubleCount = useMemo(
  () ⇒ count * 2,
  [count]
)
```

```
let [count, setCount] = useState(0)

let addToCount = useCallback(
  (value) ⇒ setCount(count + value),
  [count]
)

// somewhere later
addToCount(42)
```

⚠️ RFC

```
let [count, setCount] = useState(0)

let addToCount = useEvent(
  (value) ⇒ setCount(count + value)
)

// somewhere later
addToCount(42)
```

Memoization / Vue

```
let count = ref(0)

let doubleCount = computed(
  () ⇒ count.value * 2
  // no dependencies?
)
```

```
let count = ref(0)

function addToCount(value) {
  count.value = count.value + value
}

// somewhere later
addToCount(42)
```

Effects / React

```
let [state, setState] = useState(0)

useEffect(() ⇒ console.log(count), [count])

// somewhere later
setState(42)
```

```
useEffect(() ⇒ {
  // subscribe to something
  return () ⇒ {
    // unsubscribe from subscribed
  }
}, [/* dependencies */])
```

```
let [state, setState] = useState(0)

useLayoutEffect(
  () ⇒ console.log(count),
  [count]
)

// somewhere later
setState(42)
```

Effects / Vue

```
let count = ref(0)

watch(
  count,
  (newC, oldC) ⇒ console.log(newC),
  { flush: 'pre' } // post, sync
)

// somewhere later
count.value = 42
```

```
watchEffect((onCleanup) ⇒ {
  // subscribe to something
  onCleanup(() ⇒ {
    // unsubscribe to subscribed
  })
})
```

```
let count = ref(0)

watchEffect(
  () ⇒ console.log(count.value),
  { flush: 'pre' } // post, sync
)

// somewhere later
count.value = 42
```

Aliases

✅ **watchPostEffect**

✅ **watchSyncEffect**

Lifecycle / React

useLayoutEffect sameish?

```
useEffect(() ⇒ {
  // component mounted or updated deps
  return () ⇒ {
    // component will unmount or updated deps
  }
}, [/* deps */])
```

missing classes? 😅

Lifecycle / Vue

- **onMounted()**
- **onUpdated()**
- **onUnmounted()**
- **onBeforeMount()**
- **onBeforeUpdate()**
- **onBeforeUnmount()**
- **onActivated()**
- **onDeactivated()**
- … and 4 more hooks

```
onMounted(() => {
  // mounted, code here, not watched
})
```

```
onUnmounted(() => {
  // unmounted, code here, not watched
})
```

References / React

```
function Component() {
  let inputRef = useRef(null)

  useEffect(() => {
    // somewhere later
    inputRef.current.focus()
  })

  return <input ref={inputRef} />
}
```

References / Vue

```
let Component = defineComponent({
  setup() {
    let inputRef = ref(null)

    onMounted(() => {
      // somewhere later
      inputRef.value.focus()
    })

    return () => <input ref={inputRef} />
  }
})
```

⭐ Did we found the perfect solution for better logic reuse?

nothis, **datasync**, **composable**, **colocation**, **declarative**

Experiments

Convert React Hook to Vue Composition API (YouTube Video)
bit.ly/convert-react-to-vue

Converting @floating-ui/react-dom
github.com/allindevelopers/vue-floating-ui

Converting react-dropzone
github.com/allindevelopers/vue-use-dropzone

Images generated with
codeimage.dev

Q.A.

Be first to ask a question!
Hope 1 year of Vue will answer them

bit.ly/jsmd-hooks-to-capi