一起探索 VueUse 背後的黑魔法 🪄

IlyaL

maru VueUse 團隊成員

Unplugin 外部合作者

任職於 德安資訊

ilyal.me ilyaliao ilyaliao ilyaliao liao.0324

🙋 有聽說過 VueUse 嗎？

🙋 有用過 VueUse 嗎？

組合式函數

什麼是Composables?

來做一個簡單的 `useMouse` 吧！

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

const x = ref(0)
const y = ref(0)

function update(event) {
  x.value = event.pageX
  y.value = event.pageY
}

onMounted(() => {
  window.addEventListener('mousemove', update)
})
onUnmounted(() => {
  window.removeEventListener('mousemove', update)
})
</script>

<template>
  <div>Mouse position is at: {{ x }}, {{ y }}</div>
</template>
```

TS useMouse.ts

```ts
import { onMounted, onUnmounted, ref } from 'vue'

export function useMouse() {
  const x = ref(0)
  const y = ref(0)

  function update(event) {
    x.value = event.pageX
    y.value = event.pageY
  }

  onMounted(() => {
    window.addEventListener('mousemove', update)
  })
  onUnmounted(() => {
    window.removeEventListener('mousemove', update)
  })

  return { x, y }
}
```

還能再優化嗎？

```ts
import { ref, onMounted, onUnmounted } from 'vue'

export function useMouse() {
  const x = ref(0)
  const y = ref(0)

  function update(event) {
    x.value = event.pageX
    y.value = event.pageY
  }

  onMounted(() => {
    window.addEventListener('mousemove', update)
  })
  onUnmounted(() => {
    window.removeEventListener('mousemove', update)
  })

  return { x, y }
}
```

TS event.ts

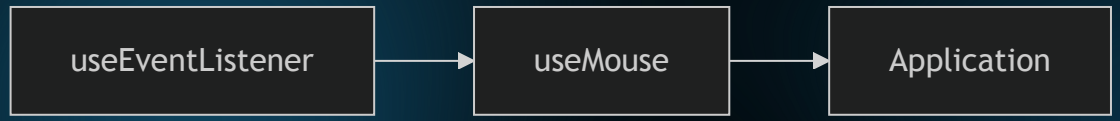
```ts
import { onMounted, onUnmounted } from 'vue'

export function useEventListener(
  target,
  event,
  callback,
) {
  onMounted(() => {
    target.addEventListener(event, callback)
  })
  onUnmounted(() => {
    target.removeEventListener(event, callback)
  })
}
```

我們做了什麼

- 將負責更新滑鼠位置的函數抽離出來 - useMouse
- 將負責綁定事件的函數抽離出來 - useEventListener
- 可以重複使用
- 關注點分離

函數關係圖



Vue 组合式 API 工具合集

v13.6.0 13M/month docs & demos 274 functions

Stars 22k

- Tree-shake
- TypeScript
- 支援 CDN
- SSR 支援 (nuxt)
- 生態系豐富

VueUse

Collection of Vue Composition Utilities

Collection of Essential Vue Composition Utilities

Get Started | Functions | Add-ons | View on GitHub

Feature Rich

200+ functions for you to choose from

Built for Vue 3

Designed for Vue 3 to take full advantage of its latest capabilities

Fully tree shakeable

Only take what you want

Type Strong

Written in TypeScript, with full TS docs

Flexible

Passing refs as arguments, fully customizable, configurable event filters and targets

No bundler required

Usable via CDN, without any bundlers

魔法師都有哪些 🪄 **工具** 呢？

🪄 MaybeRef 和 MaybeRefOrGetter

類型工具

```ts
export type MaybeRef<T = any>
  = | T
    | Ref<T>
    | ShallowRef<T>
    | WritableComputedRef<T>

export type MaybeRefOrGetter<T = any>
  = | MaybeRef<T>
    | ComputedRef<T>
    | (() ⇒ T)
```

```ts
/**
 * A ref which will be reset to the default value after some time.
 *
 * @see https://vueuse.org/shared/refAutoReset/#refautoreset
 */
import { refAutoReset } from '@vueuse/core'
import { computed, ref } from 'vue'

const rawMessage = 'default message'
const refMessage = ref(rawMessage)
const comMessage = computed(() => refMessage.value)

refAutoReset(rawMessage, 1000)
refAutoReset(refMessage, 1000)
refAutoReset(comMessage, 1000)
```

- 💡 在 Vue 3.3 中引入
- PR：https://github.com/vuejs/core/pull/7997

🪄 toValue

將響應式資料轉為純值

```ts
export function toValue<T>(source: MaybeRefOrGetter<T>): T {
  return isFunction(source) ? source() : unref(source)
}
```

```ts
import { ref, toValue } from 'vue'

const refValue = ref('123')
const value = toValue(refValue)
```

- 💡 在 Vue 3.3 中引入
- PR：https://github.com/vuejs/core/pull/7997

🪄 MaybeElement 和 MaybeComputedElementRef

類型工具

```ts
import type { ComponentPublicInstance, MaybeRef, MaybeRefOrGetter } from 'vue'
import { toValue } from 'vue'

export type VueInstance = ComponentPublicInstance
export type MaybeElement = HTMLElement | SVGElement | VueInstance | undefined | null
export type MaybeComputedElementRef<T extends MaybeElement = MaybeElement> = MaybeRefOrGetter<T>
```

```ts
import { useElementSize } from '@vueuse/core'
import { useTemplateRef } from 'vue'

const el = useTemplateRef<HTMLElement>('el')
const { width, height } = useElementSize(el)
```

- Playground：With Vue Component

🪄 unrefElement

將響應式資料轉為 DOM 元素

```ts
export function unrefElement<T extends MaybeElement>(elRef: MaybeComputedElementRef<T>): UnRefElementReturn<T> {
  const plain = toValue(elRef)
  return (plain as VueInstance)?.$el ?? plain
}
```

```ts
import type { VueInstance } from '@vueuse/core'
import { unrefElement } from '@vueuse/core'
import { useTemplateRef } from 'vue'

const elRef = useTemplateRef<HTMLElement>('el')
const el = unrefElement(elRef)

const elRef2 = useTemplateRef<VueInstance>('el')
const el2 = unrefElement(elRef2)
```

🪄 重載

類型體操 🤓

```ts
export function useToggle<Truthy, Falsy, T = Truthy | Falsy>(initialValue: Ref<T>, options?: UseToggleOptions<Truthy
export function useToggle<Truthy = true, Falsy = false, T = Truthy | Falsy>(initialValue?: T, options?: UseToggleOpt

export function useToggle(
  initialValue: MaybeRef<boolean> = false,
  options: UseToggleOptions<true, false> = {},
): UseToggleReturn {
  // ...
}
```

不同的參數，會有不同的返回的結果

```ts
import { useToggle } from '@vueuse/core'

const [isOpen, toggle] = useToggle(false)
```

魔法師都如何運用這些工具製造 ✨ 魔法

✨ useEventListener

萬用的事件綁定函數

```ts
import { tryOnMounted, tryOnUnmounted } from '@vueuse/core'

export function useEventListener(
  target: EventTarget | null | undefined,
  event: string,
  listener: (event: Event) => void,
) {
  tryOnMounted(() => {
    target.addEventListener(event, callback)
  })
  tryOnUnmounted(() => {
    target.removeEventListener(event, callback)
  })
}
```

💡 技巧

使用 tryOnMounted 和 tryOnUnmounted 來避免在元件以外的地方執行時出錯

✨ tryOnScopeDispose

安全的 onScopeDispose()

```ts
import { tryOnUnmounted } from '@vueuse/core'
import { onScopeDispose } from '@vueuse/shared'

export function useEventListener(
  target: EventTarget | null | undefined,
  event: string,
  listener: (event: Event) => void,
) {
  target?.addEventListener(event, callback)

  onScopeDispose(() => {
    target.removeEventListener(event, callback)
  })
}
```

ⓘ 資訊

RFC: https://github.com/vuejs/rfcs/blob/master/active-rfcs/0041-reactivity-effect-scope.md

✨ watchImmediate

watch 家族的成員之一

```ts
import { tryOnUnmounted } from '@vueuse/core'
import { onScopeDispose } from '@vueuse/shared'

export function useEventListener(
  target: EventTarget | null | undefined,
  event: string,
  listener: (event: Event) => void,
) {
  target?.addEventListener(event, callback)

  onScopeDispose(() => {
    target.removeEventListener(event, callback)
  })
}
```

- watchOnce
- watchDeep
- watchWithFilter
- watchTriggerable
- watchPausable
- watchIgnorable

📖 魔法書 還寫了什麼？

📖 響應式

```
import { useMouse } from './mouse.ts'

const { x, y } = useMouse()
```

←

```
const x = ref(0)
const y = ref(0)
return { x, y }
```

可以用 reactive 來包裝返回結果

```
import { useMouse } from './mouse.ts'

const mouse = reactive(useMouse())
// mouse.x 鏈接到了原來的 x ref
console.log(mouse.x)
```

📖 響應式

重複使用傳進來的響應式數據，或是轉換成成響應式數據後再返回

```ts
import { ref, shallowRef } from 'vue'

const x = ref(0)
const plusX = ref(x) // 自動解包
```

```ts
const mouse = shallowRef({ x: 0, y: 0 })
const mouse2 = ref(mouse) // ⚠️ 還是淺層
```

如何應用

```ts
export function useTitle(newTitle: MaybeRef<string>) {
  const title = ref(newTitle || document.title)

  watchImmediate(title, t => document.title = t)

  return title
}
```

→

```ts
const title = useTitle()
title.value = 'Hello World'
```

```ts
const computedTitle = computed(() => (name.value))
const title = useTitle(computedTitle)
```

📖 副作用清理

- watch
- addEventListener
- Web API
  - MutationObserver
  - IntersectionObserver
  - ResizeObserver

📖 ShallowRef

優先使用 shallowRef 而不是 ref

```
export function useFetch<T>(url: MaybeRefOrGetter<string>) {
  // use `shallowRef` to prevent deep reactivity
  const data = shallowRef<T | undefined>()
  const error = shallowRef<Error | undefined>()

  fetch(toValue(url))
    .then(r => r.json())
    .then(r => data.value = r)
    .catch(e => error.value = e)

  /* ... */
}
```

技巧

- 多餘的深層響應式會增加不必要的性能開銷

📖 Readonly

對於不應該被修改的資料，使用 readonly 來保護

```
const { width, height } = useWindowSize()

width.value = 100 // no warning
height.value = 100 // no warning
```

應該用 `readonly` 包裝後再返回

```
import { readonly } from 'vue'

return {
  width: readonly(width),
  height: readonly(height),
}
```

當嘗試修改時，會出現警告

```
const { width, height } = useWindowSize()

width.value = 100 // ⚠️ [Vue warn]
height.value = 100 // ⚠️ [Vue warn]
```

📖 getCurrentInstance

獲取當前組件實例

拿到組件實例後，就可以拿到組件的 `el`

```ts
export function useCurrentElement() {
  const vm = getCurrentInstance()!
  return vm?.proxy!.$el
}
```

拿到組件實例後，就可以拿到組件的 `refs`

```ts
export function templateRef(key: string) {
  const vm = getCurrentInstance()
  return vm?.proxy?.$refs[key]
}
```

- 💡 `getCurrentInstance` 在 `Vue 3+` 中引入
- 💡 `useTemplateRef` 在 `Vue 3.5` 中引入

我應該使用這些 🔮 **魔法** 嗎？

VueUse 能幫助我們解決什麼問題？

開箱即用
提供許多現成的組合式函數

關注點分離
將業務邏輯抽離

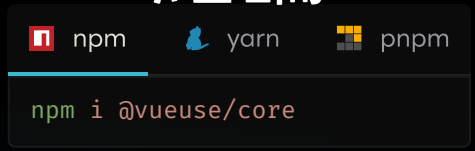
迭代更新
讓你在各個版本都能使用

最佳實踐
學習如何撰寫組合式函數

安全可靠
大量的用戶使用，穩定性有保障

生態系豐富
全家桶服務，讓你 @vueuse 打天下

結論



參考

- Vue 官方文件
- VueUse 官方文件
- VueUse 中文文檔
- 可組合的 Vue
- VueUse 最佳實踐

感謝 💚

投影片可在 ilyaliao/talks 瀏覽

靈感來自 Anthony Fu， 專案配置參考 Sxzz， 模板參考 LittleSound

簡報製作 Powered by Slidev