**COMPOSITION API (Vue 3)**

***ref()*:** thường sử dụng cho kiểu dữ liệu nguyên thủy (string, int, boollean,..)

- để truy cập vào giá trị của biến khi được khai báo bằng ref() thì ta phải .value

(VD: name.value)

- nếu biến ref() là một object thì ta vẫn phải .value mới lấy được giá trị của object đó

(VD: car.value.name)

- ta có thể thay đổi toàn bộ giá trị của object khi khai báo bằng ref() vì trong ref() có bộ set get riêng dành cho value

***reactive*** (giúp chúng ta theo dõi, thay đổi BÊN TRONG object): thường sử dụng với kiểu Mảng và Object

- truy cập vào phần tử của mảng hay object thì truy cập bình thường (VD: car.name)

- ta không thể thay đổi giá trị của object bằng reactive vì bộ set get nằm bên trong object

***computed***: khi giá trị trong hàm computed bị phụ thuộc và thay đổi thì hàm trong computed sẽ tính toán lại, **computed không có tham số truyền vào**

***watch***: sử dụng để theo dõi 1 biến nào đó -> làm cái gì đó khi 1 biến bị thay đổi

***~ watchEffect***: giống như computed dùng để tính toán. Nhưng mà **không trả về dữ liệu**

**SETUP(props, context):** 2 tham số mặc định của setup.

- Nếu không sử dụng props mà chỉ sử dụng context thì: setup(\_, context)

- props: viết trên hàm setup()

- context: đại diện cái đang ở trong component (tựa như this): có emit, attrs,...

**VUEX**

- là một parttern + library cho ứng dụng sử dụng VueJS

- là nơi để trao đổi dữ liệu giữa các component với nhau, quản lý được dữ liệu

***- getters***: mỗi getter sẽ có mỗi nhiệm vụ khác nhau

- mapGetters:

(cúpháp) computed: {

...mapGetters(["tên\_hàm\_trong\_getters","tên\_hàm\_trong\_getters",...])

}

***- mutations***: thực hiện thông qua những lần commit. giúp thay đổi giá trị trong state

truyền vào state, lấy state đó để truy xuất vào biến bên trong

***- actions:*** call API

vue component tương tác với actions thông qua dispatch

commit lên mutations

**LIFE CYCLE**

1. Creation (Initialization) – khởi tạo component

Creation là bước đầu tiên trong chuỗi Vuejs life cycle, ở bước này, chúng ta chưa có quyền truy cập vào DOM tree.

* 1. beforeCreate – trước khi tạo component

Before Create là sự kiện khởi đầu trong VueJS life cycle, tất cả component **trước khi khỏi tạo đề gọi qua beforeCreate() => beforeCreate LUÔN ĐƯỢC GỌI VÀ GỌI ĐẦU TIÊN**

* 1. Created – đã tạo ra component

ở đây ta có thể xử lý các dữ liệu trả về từ beforeCreated.

**Không thể thực hiện thao tác DOM ở bước này.**

1. Mouting (DOM insertion) – Thao tác với DOM tree

**cho phép ta truy cập tới các thành phần trên DOM TRƯỚC HOẶC SAU LẦN RENDER ĐẦU TIÊN**.

vậy nếu chúng ta cần sửa đổi, truy cập tới một phần tử trước và sau khi DOM render. Các methods nên được viết ở phần mounting, còn nếu bạn muốn set data khởi tạo cho các component, các **method fetch dữ liệu nên được viết ở created.**

* 1. beforeMount – trước khi mount

**beforeMount() cũng được gọi trước khi các điều kiện render (conditional render) trên template được gọi** => CÁC ELEMENT trên DOM chưa thể truy cập được ở BEFORE MOUNT

* 1. mounted – đã thao tác xong với DOM tree

ở event Mounted, có thể truy cập được tới các element trên DOM.

1. Updating (Diff & Re-render) – phản ánh thay đổi và re-render lại DOM tree
   1. beforeUpdate – trước khi cập nhật

beforeUpdate hooks sẽ chạy trước khi sự kiện update trên component bắt đầu. M**ột component sau khi được render lên DOM sẽ có thể được cập nhật lại.**

* 1. updated – sau khi update

trong Vuejs Life Cycle, event updated sẽ bắt đầu chạy sau khi **CÔNG VIỆC RENDER NỘI DUNG UPDATE TRÊN DOM hoàn tất**.

1. Destruction – hủy bỏ

Destruction hooks là những actions cho phép ta handle sự kiện hủy bỏ các component. **Thực hiện trước khi component được khóa khỏi cây DOM**.

* 1. beforeDestroy – trước khi hủy

Ở event này, trước khi nó được gọi, component vẫn có đầy đủ các thành phần (present) và chứng năng (functional). Nếu một event hoặc một variable **gây tốn kém về bộ nhớ** (nếu không được dọn dẹp sau khi khởi tạo), **ta có thể xử lý chúng ở đây**.

* 1. Destroyed – đã hủy component

**EVENT DESTROY CÓ THỂ DÙNG ĐỂ KIỂM CHỨNG CHO VIỆC MUỐN DESTROY CÁC ĐỐI TƯỢNG MONG MUỐN.**