|  |  |  |
| --- | --- | --- |
| Macintosh HD:Users:hoangnguyen:Documents:TDC:fitlog_blue.png | TRƯỜNG CAO ĐẲNG CÔNG NGHỆ THỦ ĐỨC  **Khoa Công Nghệ Thông Tin** | Macintosh HD:Users:hoangnguyen:Documents:TDC:logoTDC_blue.png |
|  |  |  |

**Chuyên đề phát triển web 2** | HKI – [2019 – 2020]

PRJ – REPORT

**BÁO CÁO**

**TÌM HIỂU VUE JS**

**Nhóm E:**

* **Huỳnh Thanh Duy –** 17211TT2104

Mục Lục

**Vue JS**

[I. Giới thiệu 3](#_Toc24194436)

[II. Hướng dẫn cài đặt 3](#_Toc24194437)

**Vue JS Instance**

[I. Vue Instance là gì? 4](#_Toc24194439)

[II. Vue Constructor 5](#_Toc24194440)

[III. Property and method (Thuộc tính và phương thức) 6](#_Toc24194441)

[IV. Lifecycle Hooks 8](#_Toc24194442)

**Vue Component**

[I. Cách tạo và sử dụng 13](#_Toc24194445)

**Binding**

[I. Giới thiệu 16](#_Toc24194446)

[II. Cách dùng 16](#_Toc24194447)

**Event**

[I. Giới thiệu 19](#_Toc24194448)

[II. Cách dùng 19](#_Toc24194449)

**Rendering**

[I. Giới thiệu 21](#_Toc24194450)

[II. Cách dùng 21](#_Toc24194451)

**Animation trong Vue JS**

[III. Giới thiệu 24](#_Toc24194452)

[IV. Cách dùng 24](#_Toc24194453)

Vue Js

# Giới thiệu

* Vue Js là một framework giúp cho việc xây dựng giao diện người dùng. Cốt lõi của nó chủ yêu là phần **View**.
* Vue js là một dạng **progressive** **framework** (framework linh động), cho phép và khuyến khích phát triển ứng dụng theo từng bước.
* VueJS sử dụng mô hình **MVVM** (**Model**- **View**- **ViewModel**)

# Hướng dẫn cài đặt

**Cách 1: Dùng CDN**

Đây chắc chắn là cách nhanh gọn nhất, khi bạn chỉ cần thêm 1 đường link vào là xong.



**Cách 2: Download**

Đây là cách truyền thống nhất, bạn lên trang chủ và tải file .js về rồi thêm đường dẫn tới file js đó.



### Cách 3:  NPM (Node Package Manager)

* Đây là công cụ tạo, quản lí các gói thư viện Javascript, tức là sẽ quả lí tự động, loại bỏ các thao tác thủ công như hai cách trên.
* Chỉ cần khai báo những thư viện cần sử dụng vào file **package.json**, npm sẽ sự động tải chúng về dự án.

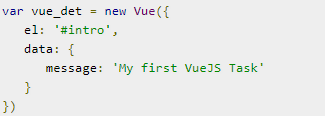
**B1:** Cài đặt Node kèm **npm**. Link: <https://nodejs.org/en/download/>

**B2:** ghõ câu lệnh **npm install vue**

Vue Js Instance

# Vue Instance là gì?

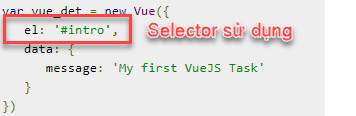
Chúng ta có thể coi Vue là một lớp có tên gọi là Vue, lúc này để khởi tạo một Instance thì bạn sẽ sử dụng cú pháp như sau:



*Cách khai báo Vue instance*

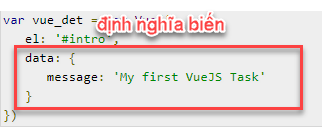
\*Trong đó:

+ **el** xác định selector nào được Vue Instance này tác động



*Xác định instance này dùng cho selector nào*

+ **data** chứa các biến của instance này



*Định nghĩa biến trong vue*

Để gọi biến này ra bạn chỉ cần gọi tên biến trong dấu ngoặc kép



*Gọi biến trong vue js*

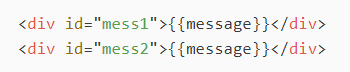
# Vue Constructor

Khi khởi tạo một Instance thì bạn phải truyền một tham số vào, và tham số này là một object chứa các thuộc tính như*: data, template, element (*el*), method* và các hàm *callback* được định nghĩa trong lifecyle.

Tạo ra 2 Instance



*Tạo 2 instance mv1, mv2*



*Gọi biến message của mỗi instance*

# Property and method (Thuộc tính và phương thức)

Tất cả các *property* và *method* bạn muốn đưa vào Vue đều phải thông qua key data lúc khởi tạo, key data này ta gọi là biến được ủy nhiệm (proxies) cho Vue instance khi vừa được khởi tạo.

*\*Ví dụ về property*



*Khai báo property*

*C:\Users\USER\AppData\Local\Microsoft\Windows\INetCache\Content.Word\1.png*

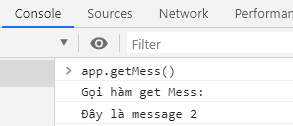
*Kết quả*

*\*Ví dụ về method*



*Khai báo method*

Kết quả:



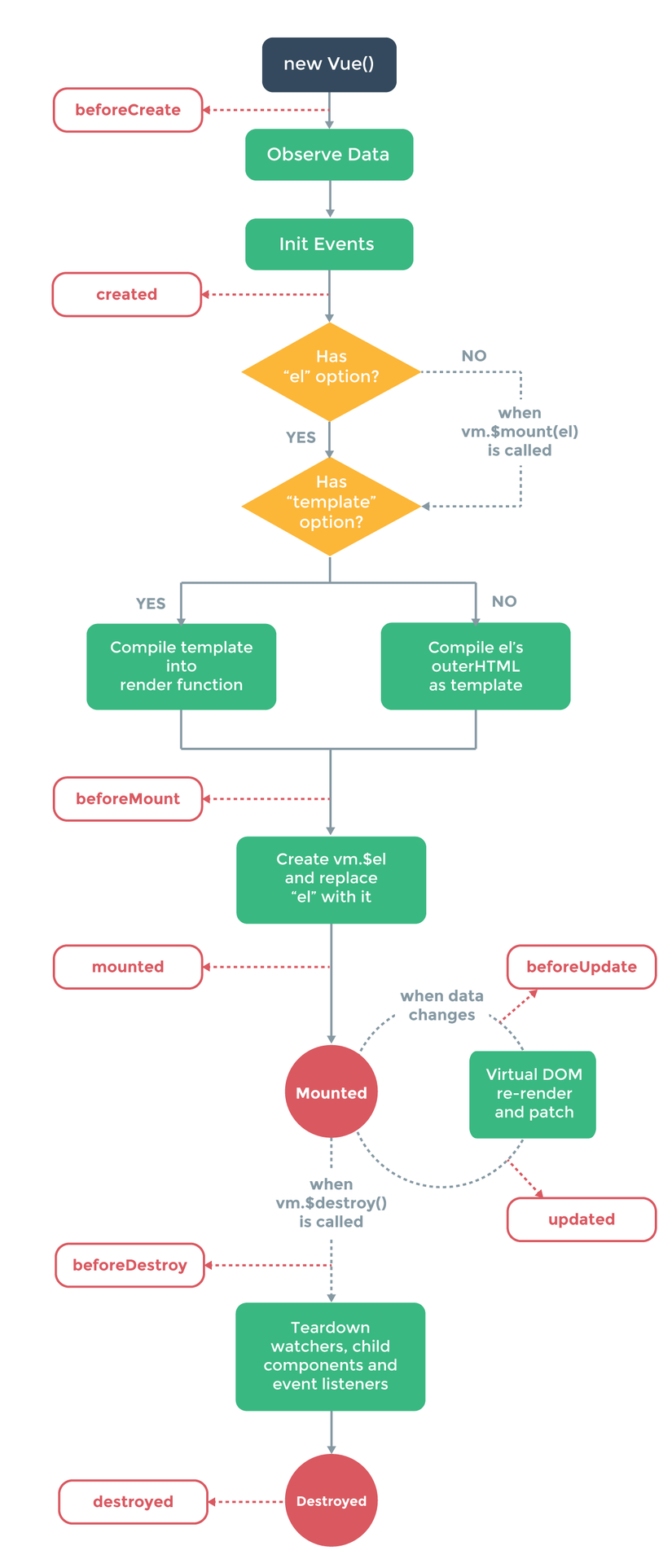
*Kết quả khi gọi hàm getMess()*

# Lifecycle Hooks

Lifecycle hooks chính là những **hàm, phương thức đặc biệt** xử lý cho một chức năng nào đó. Ví dụ khi Vue instance được khởi tạo thì chúng ta có hook tên là **created**, hook này sẽ được gọi khi instance khởi tạo xong.

Ngoài ra còn rất nhiều hooks khác như **mounted, updated, destroyed**. Tất cả các hooks này sẽ được gọi thông qua đối tượng **this** (chính là vue instance).

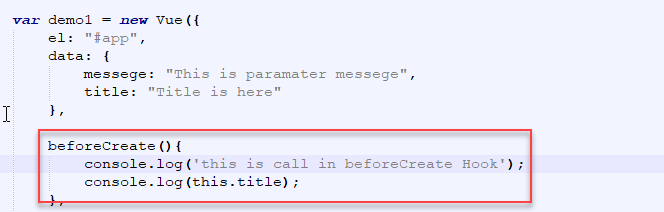
Dưới đây là một Lifecycle Diagram.



**\* beforeCreate()**

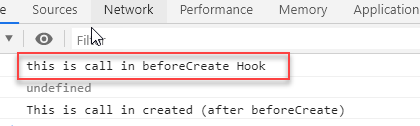
- Hook beforeCreate sẽ được gọi đồng bộ ngay **sau khi Vue được khởi tạo**. Các **data (dữ liệu) và event (sự kiện)** chưa được thiết lập.

Ví dụ với đoạn code sau:



*Sử dụng hàm beforecreate*

Kết quả:



*Biến title được gọi trước khi render ra code html*

## **Mounting (Chèn phần tử DOM)**

**Mounting hook** được sử dụng nhiều nhất. Ở giai đoạn này, Vue cho phép truy xuất vào các component ngay lập tức, sau khi component được render lần đầu tiên.

**+ Sử dụng**: Nếu muốn thay đổi DOM trước hoặc sau khi render.

**+ Không sử dụng được**: Liên quan đến dữ liệu (data).

### \* mounted()

Trong mounted hook, chúng ta có thể hoàn toàn truy cập đến component, template và DOM thông qua **this.$el** (hoặc **vm.$el**).

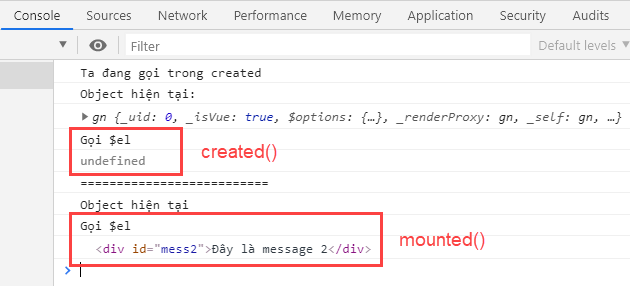
Hook này được sử dụng thường xuyên. Có thể sử dụng để t**hay đổi DOM, fetching data** ( cũng có thể dùng created() hook).

Ví dụ sau sẽ mô tả sự khác nhau giữa **created() hook** và **mounted() hook**:



*Cùng gọi this.$el ở cả created, mounted và xem kết quả*

Kết quả:



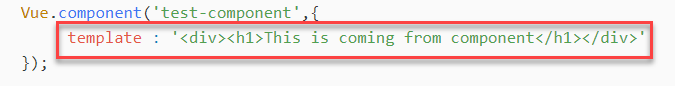
*created gọi trước khi ra mã html và mounted gọi sau khi đã ra mã*

Vue Component

Một trong những đặc trưng quan trọng nhất của Vue đó là **component**

# Cách tạo và sử dụng

1. **Cú pháp:**



*Khai báo một component*

* Ta khai báo một component có tên ‘**test-component’.**
* Key **template** chính là **nội dung code** **HTML** của component

1. **Ví dụ:**



*Gọi và khai báo component trong #app*

Kết quả:



*Đã render ra code html của component test-component*

**\* Ngoài ra ta có thể khai báo data và methods cho component**



*Có thể khai báo data và methods trong component*

**

*Khi rê chuột vào tên sẽ cập nhật*

Binding

# Giới thiệu

* Trong một thẻ HTML hầu hết đều sẽ có các class và style cho lớp đó
* Binding giúp ta thay đổi việc thêm xóa một class trong thẻ một cách linh hoạt hơn.

# Cách dùng

1. **V-bind cho class:**

Giả sử ta có class *red*:



Khai báo class .*red*

Sử dụng **v-bind:class**



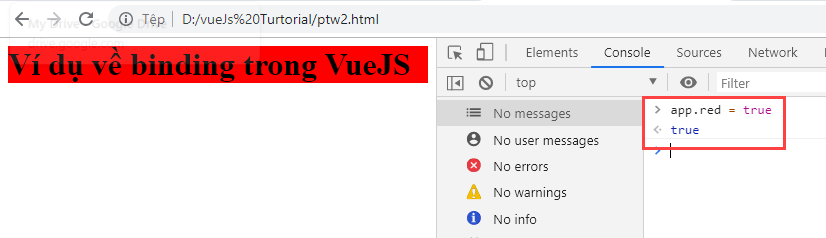
*class red sẽ thêm hoặc xóa dựa vào thuộc tính red*

Kết quả:



*Lúc này chưa có class red*

Sau khi thay đổi giá trị biến *red*:



*Lúc này class red đã được thêm vào*

1. **V-bind cho style:**



*Style sẽ phụ thuộc vào color*

Kết quả:



Event

# Giới thiệu

- Trong Vue có hỗ trợ loại sự kiện như click, mouseup, mouseout,…

- Cú pháp để sử dụng event cũng rất đơn giản. Như khi muốn gọi sự kiện click cho một button ta có cú pháp sau:



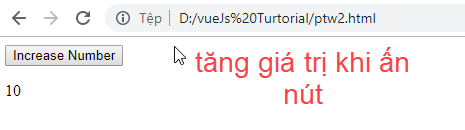
*Dùng sự kiện click cho nút*

# Cách dùng



*Khởi tạo và gọi hàm increaseNumber() cho sự kiện click của thẻ button*

Kết quả:



*Giá trị biến num sẽ tăng khi ấn button*

Rendering

# Giới thiệu

- Render là một dạng **vòng lặp, cấu trúc rẻ nhánh (if…else..)…** khi render ra code HTML khi sử dụng Vue.

# Cách dùng

1. **Vòng lặp (v-for)**

**B1:** Trước hết ta cần khai báo một biến dạng mảng trong data



*Khai báo mảng list*

**B2:** dùng vòng lặp v-for

**

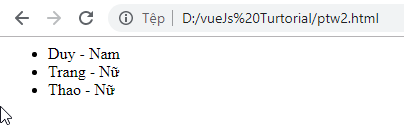
*Dùng vòng lặp để lặp thẻ li*

**B3:** đầy đủ code như sau



*Đầy đủ code*

Kết quả:



*In ra danh sách list trong vòng lặp thẻ li*

1. **Cấu trúc rẽ nhánh v-if-else**

- Giả sử tùy theo trường hợp biến mà ta sẽ render ra đoạn HTML phù hợp.

- Nếu show = 1 sẽ hiển thị thẻ h4 và là thẻ h5 nếu show = 0



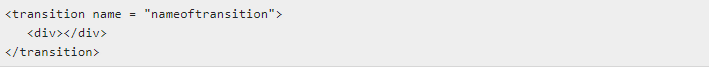
*Hiển thị thẻ h4 hoặc h5 tùy theo show*

Animation trong Vue

# Giới thiệu

* Từ các bài trước ta đã biết rằng Vue cung cấp nhiều cách để tác động tới các class, style, code HTML.
* Ngoài ra Vue còn hỗ trợ một component có sẵn hỗ trợ cho việc tạo các hiệu ứng liên quan tới transition.

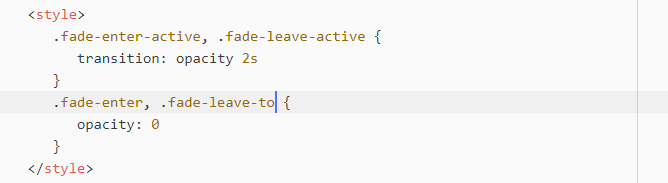
Cú pháp:



*Thẻ transitions bọc xung quanh thẻ cần transition (ở đây là div)*

# Cách dùng

**B1:** Tạo ra các style để khi gọi các sự kiện (click, hover…) sẽ dùng các class này



*Các class chứa hiệu ứng*

**B2:** Gọi thẻ transions bọc xung quanh thẻ mà ta cần cho hiệu ứng



*Sử dụng thẻ transitions (hiệu ứng dựa vào thuộc tính* ***name*** *của thẻ)*