**MỤC LỤC**

[CHƯƠNG 1. CÚ PHÁP 2](#_Toc200802556)

[1.1. GIỚI THIỆU 2](#_Toc200802557)

[1.2. Selectors 3](#_Toc200802558)

[1.3. Sepecificity 3](#_Toc200802559)

[1.4. Box Model 3](#_Toc200802560)

[CHƯƠNG 2. Layout 4](#_Toc200802561)

[2.1. Flex box 4](#_Toc200802562)

[2.2. GRID 4](#_Toc200802563)

[CHƯƠNG 3. Responsive Design 5](#_Toc200802564)

[3.1. Media queries 5](#_Toc200802565)

[3.2. Mobile first 5](#_Toc200802566)

[CHƯƠNG 4. Animation 6](#_Toc200802567)

[4.1. Keyframs 6](#_Toc200802568)

[4.2. Trasitions 6](#_Toc200802569)

[CHƯƠNG 5. Tìm hiểu về css framework 7](#_Toc200802570)

[5.1. Cơ bản về tailwindCSS. 7](#_Toc200802571)

# CÚ PHÁP

Lưu ý: Các kiến thức cần nắm

**BEM**: block element và modifier (khối thành phần và đặc điểm của khối hoặc thành phần)

**SELECTORS:** Cần học đầy đủ cả các loại:

* Basic (element, class, id)
* Combinators (descendant div p, child div > p)
* Attribute selectors [type="text"]
* Pseudo-classes (:hover, :nth-child(), v.v.)
* Pseudo-elements (::before, ::after)

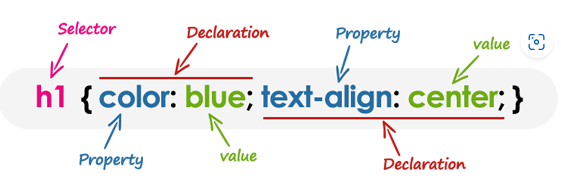
**SPECIFICITY:** Quan trọng, cần làm bài tập để hiểu rõ mức độ ưu tiên.

**BOX MODEL:** Rất cần thiết, nên bổ sung thêm các kiến thức như:

* margin collapse
* box-sizing: border-box

## GIỚI THIỆU

Cú pháp của biến định kiểu css là một tập hợp các quy tắc được web diễn giải và áp dụng cho các thẻ html.



Hình 1.1 Quy tắc đặt biến định kiểu CSS.

* CSS đều phân biệt chữ hoa chữ thường.
* Có thể lồng nhau giữa các biến định kiểu.

### Quy tắc đặt tên BEM (Block Element Modifier)

Tại sao phải dùng BEM:

Khi sử dụng css thường xuyên gặp các vấn đề như là:  
⇨ Cấu trúc html lồng nhau khi thay đổi một cấu trúc html thì css sẽ không còn hoạt động đúng như yêu cầu dẫn đến code khó bào trì và mở rộng.

.header ul li a span {

color: red;

}.

* Style bị ghi đè ngoài ý muốn. Code không có tổ chức. Phụ thuộc quá nhiều vào HTML DOM.
* Từ đó Quy tắc BEM ra đời. ngoài quy tắc BEM vẫn còn nhiều quy tắc đặt tên cho class.

Cú pháp chung của BEM là: .block\_\_element—modifier

Trong đó:

|  |  |  |
| --- | --- | --- |
| **Thành phần** | **Ý nghĩa** | **Ví dụ** |
| Block | Thành phần độc lập, có thể tái sử dụng | menu, button, form |
| Element | Thành phần con phụ thuộc vào block | menu\_\_item, form\_\_input |
| Modifier | Trạng thái, phiên bản của block hoặc element | button--active, menu\_\_item--highlighted |

Bảng 1.1 Quy tắc đặt tên BEM

Ví dụ:



Bảng 1.2 ví dụ cho bem

* Dấu cách trong thuộc tính class là biểu thị cho nhiều class như ví dụ trên thì li đầu tiên có 2 class menu\_item và menu\_item—active.

#### Lưu ý khi sử dụng BEM

👉 Đặt tên theo **ngữ nghĩa**, **chức năng**, **vai trò**, không theo màu sắc hay kích thước.

👉 Không dùng **BEM** cho những thứ nhỏ nhặt ví dụ các thẻ hr br.

👉 Chỉ sử dụng **class selectors** trong CSS.

## Selectors

**Selector** (bộ chọn) là phần dùng để **chọn ra các phần tử HTML mà bạn muốn áp dụng kiểu dáng (style)**.

### Selector cơ bản

|  |  |  |  |
| --- | --- | --- | --- |
| **Loại Selector** | **Cú pháp** | **Ý nghĩa** | **Ví dụ** |
| Phần tử | element | Chọn tất cả thẻ có tên tương ứng | p, h1, div |
| Class | .class | Chọn phần tử có class cụ thể | .menu, .title |
| ID | #id | Chọn phần tử có id cụ thể | #header, #main |
| Tất cả phần tử | \* | Chọn mọi phần tử trong tài liệu | \* { margin: 0; } |

#### Lưu ý khi sử dụng selector

### Selector kết hợp

### Pseudo-class (lớp giả)

### Pseudo-element (phần tử giả)

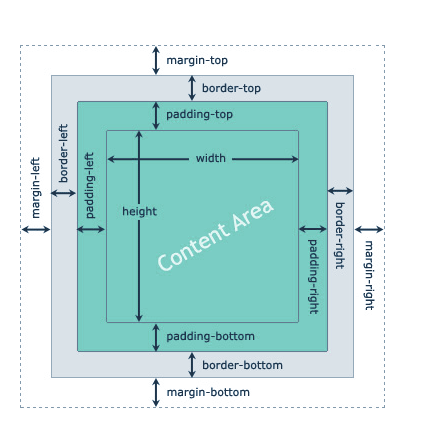
### Attribute selector (theo thuộc tính)

## Sepecificity

|  |  |  |
| --- | --- | --- |
| **Nhóm** | **Loại selector** | **Giá trị tăng thêm** |
| a | Inline style | +1000 |
| b | ID selector (#id) | +100 |
| c | Class, attribute, pseudo-class (.class, [type], :hover) | +10 |
| d | Phần tử, pseudo-element (div, p, ::before) | +1 |

Bảng 1.3 Thứ tự ưu tiên của các thành phần css

## Box Model



Hình 1.2 sơ đồ minh họa box model

# Layout

Lưu ý: Các kiến thức cần nắm

**FLEXBOX**: Rất quan trọng trong thiết kế hiện đại. Nên thực hành tạo layout đơn giản như navbar, card...

**GRID:** Khuyến khích học sau Flexbox, dùng cho layout phức tạp hơn như gallery, dashboard...

## Flex box

## GRID

# Responsive Design

Lưu ý: Các kiến thức cần nắm

**MEDIA QUERIES:** Cần học rõ cú pháp, breakpoints phổ biến.

**MOBILE FIRST:** Nên giải thích rõ triết lý "mobile-first" là gì (viết CSS cho màn hình nhỏ trước, sau đó mở rộng dần).

## Media queries

## Mobile first

# Animation

Lưu ý kiến thức cần nắm:

**KEYFRAMES:** Cần minh họa với ví dụ sinh động như loading spinner, fade in/out.

**TRANSITIONS:** Nên học rõ các thuộc tính có thể transition và cách kiểm soát timing (ease-in, ease-out, cubic-bezier, v.v.)

## Keyframs

## Trasitions

# Tìm hiểu về css framework

Lưu ý kiến thức cần nắm:

**TAILWINDCSS:** Rất phù hợp, nhưng cần đảm bảo bạn đã hiểu CSS thuần trước.

* Nên học: cấu trúc class tiện ích, cách cấu hình, responsive trong Tailwind
* Có thể mở rộng thêm Bootstrap để so sánh cách tiếp cận giữa "utility-first" và "component-based"

## Cơ bản về tailwindCSS.