**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC ĐẠI NAM**

Logo, company name

Description automatically generated

**BÀI TẬP LỚN**

**TÊN HỌC PHẦN: THIẾT KẾ WEB VÀ TRIỂN KHAI HỆ THỐNG PHẦN MỀM**

**TÊN ĐỀ TÀI: ĐỀ TÀI THIẾT KẾ TRANG WEB KHÁC BẰNG HTML VÀ CSS**

**Giảng viên hướng dẫn: ThS. Lê Văn Phong**

|  |  |  |  |
| --- | --- | --- | --- |
| **Stt** | **Mã sv** | **Họ và tên** | **Lớp** |
| **1** | **1771020256** | **Vũ Tuấn Hiệp** | **CNTT17-01** |

**Hà Nội, năm 2024**

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC ĐẠI NAM**

Logo, company name

Description automatically generated

**BÀI TẬP LỚN**

**TÊN HỌC PHẦN: THIẾT KẾ WEB VÀ TRIỂN KHAI HỆ THỐNG PHẦN MỀM**

**TÊN ĐỀ TÀI: Đề tài thiết kế các trangWeb khác bằng HTML và**

**CSS (WEB NHÀ HÀNG)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Mã sinh viên** | **Họ và tên** | **Ngày sinh** | **Điểm** | |
| **Bằng**  **số** | **Bằng**  **chữ** |
| **1** | **1771020256** | **Vũ Tuấn Hiệp** | **13/08/2005** |  |  |

**CÁN BỘ CHẤM THI**

# Hà Nội, năm 2024

Mục Lục

[CHƯƠNG I: GIỚI THIỆU VỀ THIẾT KẾ WEB VÀ TRIỂN KHAI HỆ 5](#_Toc159236993)

[THỐNG PHẦN MỀM 5](#_Toc159236994)

[1.1. Các khái niệm cơ bản 5](#_Toc159236995)

[1.2. Phân biệt Web tĩnh, Web động 7](#_Toc159236996)

[- Phân biệt Web tĩnh và Web động 7](#_Toc159236997)

[1.3. Một số thuật ngữ (Hosting, Tên miền-Domain Name-Web Server Name, HomePage, Web Site, WebServer, URL-Uniform Resource Location, Browser, …) 9](#_Toc159236998)

[1.4. Một số công cụ dùng thiết kế Web (Visual Studio code, Adobe Dreamweaver, …) 10](#_Toc159236999)

[\* Kết luận chương 1 11](#_Toc159237000)

[CHƯƠNG II: HTML & HTML5 (Hyper Text Markup Language) 13](#_Toc159237001)

[2.1. Tổng quan về HTML 13](#_Toc159237002)

[2.2. Cấu trúc tổng quát trang HTML 13](#_Toc159237003)

[2.3. Các thẻ HTML thông dụng 14](#_Toc159237004)

[2.4. Các thẻ tạo biểu mẫu (form) 16](#_Toc159237005)

[2.5. Một số thẻ HTML đặc biệt 16](#_Toc159237006)

[2.6. HTML5 18](#_Toc159237007)

[Kết luận về HTML và HTML5 22](#_Toc159237008)

[CHƯƠNG III: CSS và CSS3 (Cascading Style Sheets) 24](#_Toc159237027)

[3.1. CSS là gì? 24](#_Toc159237028)

[3.2. Cú pháp CSS 25](#_Toc159237029)

[3.3. Áp dụng CSS vào trang HTML 25](#_Toc159237030)

[3.4. Selectors 26](#_Toc159237031)

[3.4.1. Universal selector 27](#_Toc159237032)

[3.4.2. Type selector 28](#_Toc159237033)

[3.4.3. Identity selector 29](#_Toc159237034)

[3.4.4. Class selector 30](#_Toc159237035)

[3.4.5. Descendant selector 31](#_Toc159237036)

[3.4.6. Child selector Trong CSS, Child Selector cho phép bạn chọn các phần tử HTML là con trực tiếp của một phần tử khác. Nó khác với Descendant Selector ở chỗ chỉ chọn con trực tiếp, không bao gồm các cháu hoặc hậu duệ khác trong cấu trúc HTML. 32](#_Toc159237037)

[3.4.7. Adjacent sibling selector 32](#_Toc159237038)

[3.4.8. Attribute selector 33](#_Toc159237039)

[3.4.9. Pseudo class selector 34](#_Toc159237040)

[3.4.10. Group selector 35](#_Toc159237041)

[3.5. Đơn vị đo lường CSS 35](#_Toc159237042)

[3.6. Kế thừa thuộc tính 37](#_Toc159237043)

[3.7. Các nhóm thuộc tính trong CSS 38](#_Toc159237044)

[3.8. Float & Clear 40](#_Toc159237045)

[3.9. Flex 41](#_Toc159237046)

[3.10. Grid 42](#_Toc159237047)

[3.11. CSS3 42](#_Toc159237048)

[3.12. SCSS 51](#_Toc159237049)

[3.13. SASS 52](#_Toc159237050)

# 

# **CHƯƠNG I: GIỚI THIỆU VỀ THIẾT KẾ WEB VÀ TRIỂN KHAI HỆ**

# THỐNG PHẦN MỀM

## Các khái niệm cơ bản

+ Thiết kế web là gì?

     Thiết kế web là tất cả các yếu tố trực quan như màu sắc, kiểu chữ, hình ảnh và các nút điều hướng liên quan đến một trang web. Nhưng đó không đơn giản chỉ là ngoại hình thiết kế đẹp. Thiết kế web giải quyết cách bố cục một trang web, đảm bảo trang web đều hiển thị đẹp ở các kích thước màn hình khác nhau, giữ cho việc điều hướng web trở nên đơn giản và dễ dàng, đồng thời đảm bảo khả năng sử dụng của trang web trôi chảy.

+ Vai trò của thiết kế web

   Nếu coi nhà phát triển web là kỹ sư xây dựng một ngôi nhà thì nhà thiết kế web là kiến trúc sư hoặc người thiết kế nội thất. Nhà thiết kế web được coi là nghệ sĩ sáng tạo cho trang web hoặc ứng dụng, họ có trách nhiệm tập hợp các nội dung sáng tạo mang tính thương hiệu, thẩm mỹ và cung cấp giá trị cho người dùng.

   Thiết kế web và nhà thiết kế web từng là một thuật ngữ chung cho những người thiết kế trang web. Nhưng trong những năm gần đây, nhiều vai trò chuyên biệt hơn trong thiết kế web đã xuất hiện. Những vai trò này là người thiết kế UX (trải nghiệm người dùng) và UI (giao diện người dùng).



Dưới đây là một số khái niệm cơ bản về thiết kế web và triển khai hệ thống phần mềm:

Thiết kế web (Web design): Là quá trình tạo ra giao diện và trải nghiệm người dùng trên một trang web. Thiết kế web bao gồm việc lựa chọn màu sắc, font chữ, hình ảnh, cấu trúc layout và các thành phần khác để tạo ra một trang web hấp dẫn và dễ sử dụng.

Phát triển web (Web development): Là quá trình xây dựng và triển khai các ứng dụng web. Nó bao gồm việc phát triển các trang web tĩnh (static) hoặc động (dynamic), xây dựng giao diện người dùng, kết nối cơ sở dữ liệu và triển khai các chức năng tương tác.

Front-end: Là phần giao diện người dùng của một trang web hoặc ứng dụng web. Front-end phụ trách hiển thị dữ liệu và tương tác với người dùng thông qua các trình duyệt web. Các công nghệ phổ biến trong front-end là HTML (HyperText Markup Language), CSS (Cascading Style Sheets) và JavaScript.

Back-end: Là phần xử lý logic và lưu trữ dữ liệu của một trang web hoặc ứng dụng web. Back-end phụ trách nhận yêu cầu từ front-end, xử lý dữ liệu và gửi lại kết quả cho front-end. Các ngôn ngữ phổ biến trong back-end là PHP, Python, Ruby, Java, và JavaScript (Node.js), cùng với các hệ quản trị cơ sở dữ liệu như MySQL, PostgreSQL, MongoDB.

Responsive design: Là việc thiết kế và phát triển một trang web sao cho nó có thể tự động thích ứng và hiển thị tốt trên các thiết bị khác nhau như máy tính, điện thoại di động và máy tính bảng. Responsive design giúp đảm bảo trải nghiệm người dùng tốt và tăng tính tương thích của trang web trên nhiều nền tảng.

Hệ thống phần mềm (Software system): Là một tập hợp các phần mềm và các thành phần liên quan cùng với các quy trình và luồng công việc để đáp ứng một nhu cầu cụ thể. Hệ thống phần mềm có thể là một ứng dụng độc lập hoặc một phần của một hệ thống lớn hơn.

Triển khai hệ thống phần mềm (Software system deployment): Là quá trình triển khai và cài đặt hệ thống phần mềm vào môi trường thực tế. Nó bao gồm việc chuẩn bị, cấu hình và triển khai các thành phần phần mềm, cài đặt cơ sở dữ liệu, và kiểm tra tính ổn định của hệ thống.

Dưới đây là một số khái niệm cơ bản về triển khai hệ thống phần mềm:

* Cài đặt phần mềm: Cài đặt phần mềm là quá trình sao chép các tệp phần mềm vào máy tính. Quá trình này có thể khác nhau tùy thuộc vào phần mềm.
* Cấu hình cài đặt: Cấu hình cài đặt là quá trình thiết lập phần mềm để đáp ứng nhu cầu cụ thể của doanh nghiệp. Điều này có thể bao gồm việc đặt tùy chọn, tạo tài khoản người dùng và tải dữ liệu.
* Tải dữ liệu: Tải dữ liệu là quá trình thêm dữ liệu vào phần mềm. Dữ liệu này có thể đến từ một loạt các nguồn, chẳng hạn như bảng tính, cơ sở dữ liệu hoặc các hệ thống khác.

Thiết kế web và triển khai hệ thống phần mềm là hai quy trình quan trọng cần thiết để tạo và triển khai một trang web hoặc hệ thống phần mềm thành công. Bằng cách hiểu các khái niệm cơ bản của các quy trình này, bạn có thể đảm bảo rằng dự án của mình sẽ đáp ứng nhu cầu của bạn và thành công.

## Phân biệt Web tĩnh, Web động - Phân biệt Web tĩnh và Web động

Web tĩnh là loại website có nội dung cố định, không thay đổi theo thời gian.expand\_more Các trang web tĩnh được tạo ra bằng cách sử dụng các ngôn ngữ đánh dấu như HTML và CSS. Khi người dùng truy cập một trang web tĩnh, máy chủ sẽ gửi một tệp HTML tĩnh đến trình duyệt của họ.

Web động là loại website có nội dung được tạo ra động, dựa trên yêu cầu của người dùng. Các trang web động được tạo ra bằng cách sử dụng các ngôn ngữ lập trình phía máy chủ như PHP, Python và Ruby. Khi người dùng truy cập một trang web động, máy chủ sẽ thực thi một tập lệnh để tạo ra nội dung trang web.

**Dưới đây là bảng phân biệt Web tĩnh và Web động:**

|  |  |  |
| --- | --- | --- |
| Đặc điểm | Web tĩnh | Web động |
| Nội dung | Cố định | Thay đổi theo thời gian |
| Ngôn ngữ lập trình | HTML, CSS | PHP, Python, Ruby, JavaScript |
| Cách thức hoạt động | Máy chủ gửi tệp HTML tĩnh đến trình duyệt | Máy chủ thực thi tập lệnh để tạo ra nội dung trang web |
| Ứng dụng | Website giới thiệu, website brochure | Website bán hàng, blog, mạng xã hội |

**Ưu điểm của Web tĩnh:**

* Dễ dàng thiết kế và phát triển
* Tốc độ tải trang nhanh
* Chi phí thấp

**Nhược điểm của Web tĩnh:**

* Nội dung không thay đổi theo thời gian
* Khó tương tác với người dùng
* Tính năng hạn chế

**Ưu điểm của Web động:**

* Nội dung được cập nhật thường xuyênexpand\_more
* Tương tác cao với người dùngexpand\_more
* Tính năng phong phú

**Nhược điểm của Web động:**

* Phức tạp hơn để thiết kế và phát triển
* Tốc độ tải trang có thể chậm hơnexpand\_more
* Chi phí cao hơn

**Lựa chọn loại web nào phù hợp với bạn phụ thuộc vào nhu cầu của bạn.** Nếu bạn cần một website đơn giản với nội dung không thay đổi thường xuyên, web tĩnh là một lựa chọn tốt. Nếu bạn cần một website có nội dung được cập nhật thường xuyên và có nhiều tính năng tương tác, web động là lựa chọn tốt hơn.

## Một số thuật ngữ (Hosting, Tên miền-Domain Name-Web Server Name, HomePage, Web Site, WebServer, URL-Uniform Resource Location, Browser, …)

Một số thuật ngữ:

* Hosting: Dịch vụ lưu trữ dữ liệu website của bạn trên máy chủ, cho phép mọi người truy cập website của bạn trên internet.
* Tên miền (Domain Name): Tên gọi riêng biệt của website bạn, ví dụ như "[đã xoá URL không hợp lệ]". Nó hoạt động giống như địa chỉ nhà của website trên internet.
* Web Server Name: Tên máy chủ lưu trữ website của bạn, thường trùng với tên miền nhưng đôi khi có thể khác.
* HomePage: Trang chủ của website, là trang đầu tiên người dùng nhìn thấy khi truy cập website.
* Website: Tổng thể các trang web được liên kết với nhau dưới một tên miền chung.
* WebServer: Phần mềm máy tính xử lý các yêu cầu truy cập website và gửi lại nội dung tương ứng.
* URL (Uniform Resource Locator): Địa chỉ cụ thể của một tài nguyên trên internet, bao gồm tên miền, đường dẫn và tên tệp
* Browser: Trình duyệt web, phần mềm giúp người dùng truy cập và tương tác với các website, ví dụ như Chrome, Firefox, Safari.

Ngoài ra, còn một số thuật ngữ khác bạn có thể cần biết:

* Đăng ký tên miền: Quá trình mua và sở hữu một tên miền cụ thể.
* Thuê hosting: Thuê dịch vụ lưu trữ dữ liệu website trên máy chủ của nhà cung cấp hosting.
* Thiết kế web: Xây dựng giao diện và bố cục của website.
* Phát triển web: Lập trình các tính năng và chức năng của website.
* Nội dung website: Văn bản, hình ảnh, video và các nội dung khác trên website.
* Phân tích truy cập website: Theo dõi và phân tích lưu lượng truy cập website để hiểu hành vi người dùng.
* Tối ưu hóa công cụ tìm kiếm (SEO): Cải thiện thứ hạng của website trong kết quả tìm kiếm của công cụ tìm kiếm như Google.

## Một số công cụ dùng thiết kế Web (Visual Studio code, Adobe Dreamweaver, …)

1. Trình soạn thảo mã:

* Visual Studio Code: Trình soạn thảo mã miễn phí và mã nguồn mở với nhiều tính năng hỗ trợ lập trình web, như tô sáng cú pháp, tự động hoàn thành mã, gỡ lỗi, v.v.
* Sublime Text: Trình soạn thảo mã trả phí với nhiều tính năng tương tự như Visual Studio Code, giao diện đơn giản và tùy chỉnh cao.
* Atom: Trình soạn thảo mã miễn phí và mã nguồn mở, có giao diện trực quan và nhiều plugin hỗ trợ cho các ngôn ngữ lập trình khác nhau.

Atom

2. IDE (Integrated Development Environment):

* WebStorm: IDE trả phí dành cho JavaScript, cung cấp nhiều tính năng hỗ trợ phát triển web như kiểm tra lỗi, tự động hoàn thành mã, refactor code, v.v.
* PhpStorm: IDE trả phí dành cho PHP, cung cấp nhiều tính năng hỗ trợ phát triển web như kiểm tra lỗi, tự động hoàn thành mã, refactor code, v.v.
* IntelliJ IDEA: IDE trả phí hỗ trợ nhiều ngôn ngữ lập trình khác nhau, bao gồm Java, JavaScript, Python, PHP, v.v.

3. Công cụ thiết kế giao diện:

* Adobe Photoshop: Phần mềm chỉnh sửa ảnh chuyên nghiệp, có thể sử dụng để thiết kế giao diện web.

Adobe Photoshop

* Adobe Illustrator: Phần mềm thiết kế đồ họa vector, có thể sử dụng để thiết kế logo, icon, v.v. cho website.

Adobe Illustrator

* Sketch: Phần mềm thiết kế giao diện web chuyên nghiệp, có nhiều tính năng hỗ trợ tạo giao diện web đẹp mắt và dễ sử dụng.

4. CMS (Content Management System):

* WordPress: CMS phổ biến nhất hiện nay, dễ sử dụng và có nhiều plugin hỗ trợ mở rộng chức năng.
* Joomla: CMS mạnh mẽ và linh hoạt, phù hợp cho các website lớn và phức tạp.
* Drupal: CMS bảo mật cao và có nhiều tính năng hỗ trợ phát triển website, phù hợp cho các website cộng đồng.

Ngoài ra, còn có nhiều công cụ khác hỗ trợ thiết kế web như:

* Công cụ kiểm tra tốc độ website: Giúp bạn kiểm tra tốc độ tải trang web và đưa ra các đề xuất cải thiện.
* Công cụ tối ưu hóa SEO: Giúp bạn tối ưu hóa website để tăng thứ hạng trong kết quả tìm kiếm của Google.
* Công cụ quản lý dự án: Giúp bạn quản lý các công việc và tiến độ của dự án thiết kế web.

Lựa chọn công cụ phù hợp để thiết kế web phụ thuộc vào nhu cầu và kỹ năng của bạn.

## \* Kết luận chương 1

Thiết kế web và triển khai hệ thống phần mềm là hai quy trình quan trọng và liên quan mật thiết với nhau. Thiết kế web là quá trình tạo ra giao diện và bố cục của website, trong khi triển khai hệ thống phần mềm là quá trình cài đặt và cấu hình phần mềm để sử dụng.

Để thiết kế và triển khai một website thành công, bạn cần:

* Xác định mục tiêu và yêu cầu của website.
* Lựa chọn công cụ phù hợp.
* Thuê hosting và đăng ký tên miền.
* Thiết kế giao diện và bố cục của website.
* Lập trình các tính năng và chức năng của website.
* Cài đặt và cấu hình phần mềm.
* Kiểm tra và bảo trì website.

Thiết kế web và triển khai hệ thống phần mềm là một lĩnh vực rộng lớn với nhiều khía cạnh cần phải xem xét. Để có được một website đẹp mắt, hiệu quả và đáp ứng nhu cầu của bạn, bạn nên tham khảo ý kiến của các chuyên gia thiết kế web và lập trình viên.

Dưới đây là một số kết luận quan trọng về thiết kế web và triển khai hệ thống phần mềm:

* Website là một công cụ quan trọng cho doanh nghiệp trong thời đại công nghệ số.
* Thiết kế web và triển khai hệ thống phần mềm cần được thực hiện một cách chuyên nghiệp để đảm bảo hiệu quả.
* Có nhiều công cụ và dịch vụ khác nhau hỗ trợ thiết kế web và triển khai hệ thống phần mềm.
* Bạn nên tham khảo ý kiến của các chuyên gia để có được một website đẹp mắt, hiệu quả và đáp ứng nhu cầu của bạn.

# CHƯƠNG II: HTML & HTML5 (Hyper Text Markup Language)

## 2.1. Tổng quan về HTML

HTML (viết tắt của HyperText Markup Language) là ngôn ngữ đánh dấu siêu văn bản được sử dụng để tạo ra các trang web. Nó là một ngôn ngữ mã hóa đơn giản, sử dụng các thẻ để định dạng văn bản, hình ảnh, video và các nội dung khác trên website.

Cấu trúc cơ bản của một trang HTML:

* DOCTYPE: Khai báo phiên bản HTML của trang web.
* html: Thẻ bắt đầu và kết thúc của trang HTML.
* head: Chứa thông tin về trang web, như tiêu đề, mô tả, liên kết đến các tập tin CSS và JavaScript.
* body: Chứa nội dung chính của trang web, như văn bản, hình ảnh, video.

Các thẻ HTML phổ biến:

* Thẻ heading: Định dạng văn bản thành các tiêu đề h1, h2, h3, v.v.
* Thẻ paragraph: Định dạng văn bản thành các đoạn văn.
* Thẻ image: Chèn hình ảnh vào trang web.
* Thẻ link: Tạo liên kết đến các trang web khác.
* Thẻ table: Tạo bảng dữ liệu.
* Thẻ form: Tạo các biểu mẫu để người dùng nhập dữ liệu.

## 2.2. Cấu trúc tổng quát trang HTML

1. Doctype:

* Khai báo phiên bản HTML của trang web.
* Dòng đầu tiên của trang HTML.

2. Khai báo HTML:

* Bắt đầu và kết thúc trang HTML.
* Bao gồm thẻ html, head và body.

3. Phần head:

* Chứa thông tin về trang web, không hiển thị trực tiếp.
* Bao gồm:
  + Tiêu đề trang (title)
  + Mô tả trang (meta)
  + Liên kết đến tập tin CSS (link)
  + Kịch bản JavaScript (script)

4. Phần body:

* Chứa nội dung chính của trang web, hiển thị trực tiếp.
* Bao gồm:
  + Văn bản (p)
  + Tiêu đề (h1, h2, h3, ...)
  + Hình ảnh (img)
  + Liên kết (a)
  + Bảng (table)
  + Biểu mẫu (form)

5. Kết thúc:

* Thẻ </html> để kết thúc trang HTML.

Lưu ý:

* Doctype, thẻ HTML, head và body là bắt buộc.
* Có thể thêm nhiều thẻ khác vào phần head và body.
* Cấu trúc HTML phải được viết đúng cú pháp.

## 2.3. Các thẻ HTML thông dụng

- Các thẻ xử lý văn bản

* Thẻ heading: Định dạng văn bản thành các tiêu đề h1, h2, h3, v.v.
* Thẻ paragraph: Định dạng văn bản thành các đoạn văn.
* Thẻ strong: In đậm văn bản.
* Thẻ em: In nghiêng văn bản.
* Thẻ mark: Tô sáng văn bản.
* Thẻ small: Thu nhỏ kích thước văn bản.
* Thẻ sup: Viết văn bản lên cao.
* Thẻ sub: Viết văn bản xuống thấp.
* Thẻ br: Thêm dấu xuống dòng.
* Thẻ hr: Thêm một đường kẻ ngang.
* Thẻ p: Thêm một đoạn văn mới.
* Thẻ blockquote: Trích dẫn một đoạn văn bản.
* Thẻ pre: Giữ nguyên định dạng văn bản.

- Các thẻ tạo bảng

* Thẻ table: Tạo bảng dữ liệu.
* Thẻ tr: Định dạng từng hàng trong bảng.
* Thẻ td: Định dạng từng ô trong bảng.

- Thẻ liên kết

* Thẻ a: Tạo liên kết đến các trang web khác hoặc các vị trí khác trong cùng trang web.

- Các thẻ đa phương tiện

* Thẻ img: Chèn hình ảnh vào trang web.
* Thẻ <video> được sử dụng để nhúng video vào trang web.
* Thẻ <audio> được sử dụng để nhúng âm thanh vào trang web.

- Thẻ tạo khung

* **Thẻ** <frameset> được sử dụng để chia cửa sổ trình duyệt thành các khung (frame) con. Mỗi khung con hiển thị một trang web riêng biệt.
* Thẻ <frame>: Xác định URL của trang web hiển thị trong khung con.
* Thẻ <noframes>: Hiển thị nội dung thay thế cho người dùng không sử dụng trình duyệt hỗ trợ khung.

## 2.4. Các thẻ tạo biểu mẫu (form)

1. Thẻ form:

* Dùng để tạo một biểu mẫu.

2. Thẻ input:

* Dùng để tạo các trường nhập dữ liệu trong biểu mẫu.

3. Thẻ textarea:

* Dùng để tạo một ô nhập văn bản nhiều dòng.

4. Thẻ select:

* Dùng để tạo danh sách lựa chọn.

5. Thẻ label:

* Dùng để gắn nhãn cho các trường dữ liệu.

## 2.5. Một số thẻ HTML đặc biệt

**1. Thẻ** <DOCTYPE>**:**

* Khai báo phiên bản HTML của trang web.
* Dòng đầu tiên của trang HTML.
* Ví dụ:

**2. Thẻ** <html>**:**

* Bắt đầu và kết thúc trang HTML.
* Bao gồm thẻ head và body.
* Ví dụ:

**3. Thẻ** <head>**:**

* Chứa thông tin về trang web, không hiển thị trực tiếp.
* Bao gồm:
  + Tiêu đề trang (title)
  + Mô tả trang (meta)
  + Liên kết đến tập tin CSS (link)
  + Kịch bản JavaScript (script)

**4. Thẻ** <body>**:**

* Chứa nội dung chính của trang web, hiển thị trực tiếp.
* Bao gồm:
  + Văn bản (p)
  + Tiêu đề (h1, h2, h3, ...)
  + Hình ảnh (img)
  + Liên kết (a)
  + Bảng (table)
  + Biểu mẫu (form)

5. Thẻ <meta>:

* Cung cấp thông tin về trang web cho trình duyệt và các công cụ tìm kiếm.

6. Thẻ <link>:

* Liên kết đến các tập tin CSS và JavaScript.

**7.** Thẻ <img>:

* Chèn hình ảnh vào trang web

8. Thẻ <a>:

* Tạo liên kết đến các trang web khác hoặc các vị trí khác trong cùng trang web.

9. Thẻ <table>:

* Tạo bảng dữ liệu.

10. Thẻ <tbody>:

* Định dạng nội dung của bảng.

Lưu ý:

* Mỗi thẻ HTML có thẻ mở và thẻ đóng.
* Thẻ mở có dạng <tên\_thẻ>.
* Thẻ đóng có dạng </tên\_thẻ>.
* Một số thẻ không cần thẻ đóng, ví dụ như thẻ <br> và <img>.

## 2.6. HTML5

+ HTML5 là gì?

HTML5 là phiên bản thứ 5 của ngôn ngữ HTML, được sử dụng để cấu trúc và trình bày nội dung cho World Wide Web (WWW). Nó được xem là một bước tiến lớn trong sự phát triển của web với nhiều tính năng mới và cải tiến so với các phiên bản HTML trước đây.

+ Cú pháp HTML5

Cú pháp HTML5 có một số điểm khác biệt so với HTML4 và XHTML. Dưới đây là một số điểm chính:

**1. Doctype:**

Doctype trong HTML5 đơn giản hơn nhiều so với HTML4 và XHTML.

**2. Mã hóa ký tự:**

HTML5 sử dụng mã hóa ký tự UTF-8. Bạn nên khai báo mã hóa ký tự trong phần tử head của trang web

**3. Thẻ script:**

Thẻ script trong HTML5 có thể được sử dụng để tải tập tin JavaScript.

**4. Thẻ link:**

Thẻ link trong HTML5 được sử dụng để liên kết đến các tập tin CSS.

**5. Các phần tử HTML5:**

HTML5 có một số phần tử mới không có trong HTML4. Một số phần tử mới phổ biến nhất bao gồm:

* header: Phần đầu trang web
* section: Một phần của nội dung trang web
* article: Một bài báo hoặc bài đăng trên blog
* aside: Thanh bên lề của trang web
* footer: Chân trang web

**6. Thuộc tính** class **và `id:**

Thuộc tính class và id được sử dụng để thêm các lớp và ID vào các phần tử HTML. Lớp và ID có thể được sử dụng để định kiểu các phần tử bằng CSS hoặc để truy cập các phần tử bằng JavaScript.

**7. Thuộc tính `data-\*:**

Thuộc tính data-\* được sử dụng để lưu trữ dữ liệu tùy chỉnh trong các phần tử HTML. Dữ liệu này có thể được truy cập bằng JavaScript.

**8. Cú pháp rút gọn:**

HTML5 cho phép bạn sử dụng cú pháp rút gọn cho một số phần tử.

**9. Khả năng tương thích:**

HTML5 được thiết kế để tương thích với các trình duyệt web hiện đại. Tuy nhiên, một số trình duyệt cũ có thể không hỗ trợ tất cả các tính năng của HTML5.

+ Khai báo DOCTYPE trong HTML5

Khai báo DOCTYPE là dòng đầu tiên của một tài liệu HTML5. Nó cho trình duyệt web biết rằng tài liệu này là một tài liệu HTML5 và tuân theo các quy tắc của HTML5.

Cú pháp của khai báo DOCTYPE:

HTML

<!DOCTYPE html>

+ Thẻ ngữ nghĩa

Thẻ ngữ nghĩa (semantic element) trong HTML5 là các thẻ được sử dụng để xác định ý nghĩa và cấu trúc của nội dung trang web. Việc sử dụng thẻ ngữ nghĩa giúp cho trình duyệt web và các công cụ khác hiểu rõ hơn về nội dung trang web, từ đó giúp cải thiện khả năng truy cập, khả năng lập chỉ mục và khả năng sử dụng của trang web.

**Dưới đây là một số thẻ ngữ nghĩa phổ biến trong HTML5:**

* article**:** Dùng để xác định một bài viết độc lập, như bài báo, bài blog, v.v.
* aside**:** Dùng để xác định nội dung bổ sung cho nội dung chính của trang web, như thanh bên, ghi chú, v.v.
* details**:** Dùng để xác định nội dung chi tiết có thể thu gọn hoặc mở rộng.
* figure**:** Dùng để xác định một hình ảnh và chú thích của hình ảnh.
* footer**:** Dùng để xác định phần chân trang của trang web.
* header**:** Dùng để xác định phần đầu trang của trang web.
* main**:** Dùng để xác định nội dung chính của trang web.
* nav**:** Dùng để xác định một nhóm liên kết điều hướng.
* section**:** Dùng để xác định một phần của nội dung trang web.

**Sử dụng thẻ ngữ nghĩa mang lại nhiều lợi ích, bao gồm:**

* **Cải thiện khả năng truy cập:** Thẻ ngữ nghĩa giúp cho các công cụ hỗ trợ người khuyết tật hiểu rõ hơn về nội dung trang web, từ đó giúp họ truy cập trang web dễ dàng hơn.
* **Cải thiện khả năng lập chỉ mục:** Thẻ ngữ nghĩa giúp cho công cụ tìm kiếm hiểu rõ hơn về nội dung trang web, từ đó giúp trang web được lập chỉ mục tốt hơn.
* **Cải thiện khả năng sử dụng:** Thẻ ngữ nghĩa giúp cho người dùng dễ dàng hiểu và sử dụng trang web hơn.

+ Webform

Webform là một phần quan trọng của bất kỳ trang web nào cho phép người dùng tương tác và gửi thông tin. HTML5 cung cấp nhiều tính năng mới và cải tiến giúp việc tạo webform dễ dàng và linh hoạt hơn.

Dưới đây là một số điểm chính về webform trong HTML5:

1. Thẻ <form>:

* Đây là thẻ chính để tạo webform. Bên trong thẻ này, bạn sẽ thêm các thẻ khác để tạo các trường nhập liệu, nút submit, v.v.
* Thuộc tính action của thẻ <form> xác định nơi dữ liệu của webform sẽ được gửi tới (thường là một script xử lý trên server).
* Thuộc tính method xác định phương thức gửi dữ liệu (GET hoặc POST).

2. Các loại trường nhập liệu:

* HTML5 cung cấp nhiều loại trường nhập liệu khác nhau, chẳng hạn như:
  + text: Cho phép người dùng nhập văn bản.
  + password: Cho phép người dùng nhập mật khẩu (ký tự sẽ được ẩn khi nhập).
  + email: Cho phép người dùng nhập địa chỉ email.
  + number: Cho phép người dùng nhập số.
  + date: Cho phép người dùng chọn ngày.
  + checkbox: Cho phép người dùng chọn một hoặc nhiều giá trị.
  + radio: Cho phép người dùng chọn một trong số các giá trị.
  + textarea: Cho phép người dùng nhập văn bản đa dòng.
* Mỗi loại trường nhập liệu có các thuộc tính riêng để tùy chỉnh hành vi và giao diện.

3. Thuộc tính mới:

* HTML5 giới thiệu nhiều thuộc tính mới cho các thẻ liên quan đến webform, chẳng hạn như:
  + required: Làm cho trường nhập liệu bắt buộc phải điền.
  + placeholder: Hiển thị văn bản gợi ý trong trường nhập liệu khi trống.
  + pattern: Xác định một biểu thức chính quy để hạn chế loại dữ liệu người dùng có thể nhập.
  + min và max: Xác định giá trị tối thiểu và tối đa cho trường nhập liệu.

4. Kiểm tra dữ liệu:

* HTML5 cung cấp các thuộc tính và API để kiểm tra dữ liệu người dùng nhập vào trước khi gửi webform. Điều này giúp đảm bảo dữ liệu hợp lệ và giảm thiểu lỗi trên server.

5. Cải thiện khả năng truy cập:

* HTML5 chú trọng đến khả năng truy cập, và các thẻ và thuộc tính liên quan đến webform cũng được thiết kế để đảm bảo người dùng khuyết tật có thể sử dụng chúng dễ dàng.

\* Kết luận chương 2

## Kết luận về HTML và HTML5

**HTML (HyperText Markup Language)** là ngôn ngữ lập trình cơ bản để tạo trang web. HTML sử dụng các thẻ để xác định cấu trúc và nội dung của trang web.

**HTML5 là phiên bản mới nhất của HTML**, được phát hành vào năm 2014. HTML5 cung cấp nhiều tính năng mới và cải tiến so với HTML, bao gồm:

* **Cấu trúc đơn giản và dễ học:** HTML5 sử dụng cú pháp đơn giản và dễ hiểu hơn so với HTML4.
* **Hỗ trợ đa phương tiện mạnh mẽ:** HTML5 tích hợp sẵn các tính năng hỗ trợ đa phương tiện như video, âm thanh, đồ họa vector, canvas, v.v.
* **Khả năng tương tác cao:** HTML5 cung cấp nhiều API mới cho phép tạo các ứng dụng web tương tác và có khả năng phản hồi cao.
* **Tương thích với nhiều thiết bị:** HTML5 được thiết kế để hoạt động tốt trên nhiều thiết bị khác nhau, từ máy tính để bàn đến điện thoại thông minh và máy tính bảng.
* **Hiệu suất cao:** HTML5 được tối ưu hóa cho hiệu suất cao, giúp cho trang web tải nhanh hơn và mượt mà hơn.

**+ So sánh HTML và HTML5:**

|  |  |  |
| --- | --- | --- |
| Tính Năng | HTML | HTML5 |
| Cấu trúc | Phức tạp | Đơn giản |
| Hỗ trợ đa phương tiện | Yếu | Mạnh mẽ |
| Khả năng tương tác | Thấp | Cao |
| Tương thích với nhiều thiết bị | Kém | Tốt |
| Hiệu suất | Thấp | Cao |

HTML5 là phiên bản nâng cấp mạnh mẽ và hiện đại của HTML. Việc sử dụng HTML5 mang lại nhiều lợi ích cho việc phát triển trang web, giúp cho trang web có khả năng tương tác cao, hiệu suất tốt và tương thích với nhiều thiết bị. Do đó, **HTML5 là lựa chọn tốt nhất cho việc tạo trang web hiện nay.**

# CHƯƠNG III: CSS và CSS3 (Cascading Style Sheets)

## 3.1. CSS là gì?

CSS là viết tắt của Cascading Style Sheets, là một ngôn ngữ lập trình được sử dụng để định kiểu cho các trang web. CSS giúp bạn kiểm soát giao diện của trang web, bao gồm:

* Màu sắc
* Phông chữ
* Kích thước
* Vị trí
* Bố cục
* Hoạt ảnh
* v.v.

Cách hoạt động của CSS:

* CSS được sử dụng kết hợp với HTML. HTML được sử dụng để xác định cấu trúc và nội dung của trang web, trong khi CSS được sử dụng để định kiểu cho nội dung đó.
* CSS sử dụng các quy tắc để xác định cách thức hiển thị các phần tử HTML. Các quy tắc này bao gồm một bộ chọn (selector) xác định phần tử HTML nào sẽ được áp dụng kiểu, và một khai báo (declaration) xác định kiểu cụ thể sẽ được áp dụng.
* CSS có thể được áp dụng cho các phần tử HTML riêng lẻ, nhóm phần tử hoặc tất cả các phần tử trên trang web.

Lợi ích của việc sử dụng CSS:

* Giúp trang web đẹp mắt và thu hút hơn.
* Giúp trang web dễ sử dụng và điều hướng hơn.
* Giúp cải thiện khả năng truy cập của trang web.
* Giúp giảm thời gian tải trang web.
* Giúp tách biệt nội dung và giao diện, giúp cho việc quản lý trang web dễ dàng hơn.

Có ba loại CSS chính:

* CSS nội tuyến: được nhúng trực tiếp vào thẻ HTML.
* CSS nhúng: được đặt trong phần <head> của tài liệu HTML.
* CSS liên kết: được liên kết đến từ tài liệu HTML bằng một thẻ <link>.

## 3.2. Cú pháp CSS

Cú pháp CSS bao gồm ba phần chính:

1. Bộ chọn (Selector): Xác định phần tử HTML nào sẽ được áp dụng kiểu. Có nhiều loại bộ chọn khác nhau, bao gồm:

* Bộ chọn theo thẻ: Chọn tất cả các phần tử HTML có tên thẻ cụ thể. Ví dụ: p chọn tất cả các đoạn văn.
* Bộ chọn theo class: Chọn tất cả các phần tử HTML có class cụ thể. Ví dụ: .button chọn tất cả các phần tử có class button.
* Bộ chọn theo id: Chọn phần tử HTML có id cụ thể. Ví dụ: #header chọn phần tử có id header.
* Bộ chọn con: Chọn tất cả các phần tử con của một phần tử cụ thể. Ví dụ: div p chọn tất cả các đoạn văn bên trong các div.
* Bộ chọn anh chị em: Chọn tất cả các phần tử anh chị em liền kề của một phần tử cụ thể. Ví dụ: h1 + p chọn tất cả các đoạn văn ngay sau thẻ h1.

2. Thuộc tính (Property): Xác định kiểu cụ thể sẽ được áp dụng cho phần tử. Ví dụ: color, font-size, margin, padding, background-color, v.v.

3. Giá trị (Value): Xác định giá trị cụ thể cho thuộc tính. Ví dụ: red, 16px, 10px, 20px, #ffffff, v.v.

## 3.3. Áp dụng CSS vào trang HTML

Có ba cách chính để áp dụng CSS vào trang HTML:

1. CSS nội tuyến:

* CSS nội tuyến được nhúng trực tiếp vào thẻ HTML bằng thuộc tính style.
* Cách này hữu ích cho việc định kiểu cho các phần tử riêng lẻ, nhưng không phù hợp cho việc định kiểu chung cho toàn bộ trang web.

2. CSS nhúng:

* CSS nhúng được đặt trong phần <head> của tài liệu HTML bằng thẻ <style>.
* Cách này phù hợp cho việc định kiểu chung cho toàn bộ trang web.

3. CSS liên kết:

* CSS liên kết được liên kết đến từ tài liệu HTML bằng một thẻ <link>.
* Cách này phù hợp cho việc sử dụng cùng một tệp CSS cho nhiều trang web.

## 3.4. Selectors

CSS Selectors là những công cụ cho phép bạn nhắm mục tiêu tới các phần tử HTML cụ thể để áp dụng các kiểu CSS cho chúng. Nó giống như là bạn đang chọn các phần tử này trên trang web và sau đó tùy chỉnh giao diện của chúng bằng bộ quy tắc CSS.

Hãy hình dung việc tô màu cho một bức tranh. Thay vì tô toàn bộ bức tranh, bạn có thể tô từng phần riêng biệt bằng cách chọn chúng ra. CSS Selectors cũng hoạt động theo cách tương tự, giúp bạn chọn các phần tử HTML riêng lẻ để định dạng.

Lợi ích của việc sử dụng CSS Selectors:

* Kiểm soát chính xác cách hiển thị trang web: Bạn có thể tạo ra giao diện nhất quán và đẹp mắt cho từng phần tử trên trang web của mình.
* Tăng hiệu quả và tiết kiệm thời gian: Thay vì viết các kiểu CSS lặp đi lặp lại cho nhiều phần tử giống nhau, bạn có thể sử dụng một Selector để áp dụng kiểu cho tất cả chúng cùng một lúc.
* Cải thiện khả năng truy cập: Bạn có thể sử dụng Selectors để tạo các kiểu riêng cho các phần tử có vai trò ngữ nghĩa khác nhau (ví dụ như tiêu đề, đoạn văn, nút bấm). Điều này giúp trang web dễ hiểu hơn cho người dùng, bao gồm cả người dùng sử dụng công nghệ hỗ trợ.

Các loại CSS Selectors phổ biến:

* Type Selector: Chọn các phần tử HTML dựa trên loại thẻ của chúng (ví dụ: p, div, button).
* ID Selector: Chọn một phần tử HTML duy nhất có thuộc tính id cụ thể.
* Class Selector: Chọn các phần tử HTML có thuộc tính class cụ thể.
* Descendant Selector: Chọn các phần tử HTML là con của một phần tử khác.
* Pseudo-class Selector: Chọn các phần tử HTML dựa trên trạng thái của chúng (ví dụ: :hover, :active).

Ngoài ra, còn nhiều loại Selectors khác với độ phức tạp cao hơn, cho phép bạn chọn các nhóm phần tử dựa trên nhiều điều kiện khác nhau.

### 3.4.1. Universal selector

Trong CSS, Universal Selector (ký hiệu \*) là một kiểu Selector đặc biệt cho phép bạn chọn tất cả các phần tử HTML trên trang web. Nó giống như một phím tắt để áp dụng các kiểu cho mọi thứ cùng một lúc.

Mặc dù Universal Selector rất tiện lợi trong một số trường hợp, nhưng việc sử dụng nó thường không được khuyến khích do một số lý do:

* Thiếu tính cụ thể: Vì nó chọn tất cả phần tử, rất khó để kiểm soát chính xác các kiểu được áp dụng ở đâu. Điều này có thể dẫn đến giao diện không nhất quán hoặc các vấn đề về bố cục.
* Hiệu suất chậm hơn: Khi trình duyệt cần áp dụng các kiểu cho mọi phần tử, nó sẽ tốn nhiều thời gian hơn so với chỉ áp dụng cho các phần tử được chọn lọc kỹ lưỡng.
* Khó maintain: Khi bạn cần thay đổi kiểu cho một phần tử cụ thể, bạn phải tìm kiếm tất cả các vị trí sử dụng Universal Selector và cập nhật chúng, điều này tốn nhiều thời gian và dễ mắc lỗi.

Tuy nhiên, có một số trường hợp sử dụng Universal Selector hợp lý:

* Áp dụng một phong cách cơ bản chung cho tất cả các phần tử: Ví dụ, bạn có thể sử dụng Universal Selector để đặt màu mặc định cho văn bản hoặc khoảng cách mặc định giữa các phần tử.
* Xóa kiểu mặc định của trình duyệt: Một số trình duyệt áp dụng các kiểu mặc định cho các phần tử nhất định. Bạn có thể sử dụng Universal Selector để xóa tất cả các kiểu mặc định này và bắt đầu từ một trạng thái "trống".

Thay vì sử dụng Universal Selector, nên ưu tiên sử dụng các Selectors khác có tính cụ thể cao hơn như:

* Type Selectors: Chọn các phần tử dựa trên loại thẻ của chúng (ví dụ: p, div, button).
* ID Selectors: Chọn một phần tử duy nhất có thuộc tính id cụ thể.
* Class Selectors: Chọn các phần tử có thuộc tính class cụ thể.
* Descendant Selectors: Chọn các phần tử là con của một phần tử khác.

Bằng cách sử dụng các Selectors cụ thể hơn, bạn có thể tạo ra giao diện nhất quán, có thể bảo trì và hiệu quả hơn cho trang web của mình.

### 3.4.2. Type selector

Trong CSS, **Type Selector** là một cách đơn giản để nhắm mục tiêu tới các phần tử HTML dựa trên **loại thẻ** của chúng. Ví dụ, bạn có thể sử dụng Type Selector để định dạng tất cả các đoạn văn (p), các tiêu đề (h1, h2, h3,...), các nút bấm (button), các hình ảnh (img), v.v.

**Cách sử dụng Type Selector:**

Để sử dụng Type Selector, bạn chỉ cần viết tên của thẻ HTML trong bộ quy tắc CSS của bạn. Ví dụ, để thay đổi màu sắc của tất cả các đoạn văn trên trang web thành màu đỏ, bạn có thể sử dụng:

**Lợi ích của Type Selector:**

* **Đơn giản và dễ sử dụng:** Đây là cách dễ nhất để chọn một nhóm lớn các phần tử cùng loại.
* **Hiệu quả:** Trình duyệt có thể nhanh chóng xác định các phần tử phù hợp với Type Selector.
* **Kiểm soát toàn diện:** Bạn có thể định dạng tất cả các phần tử thuộc một loại nhất định trên trang web.

**Lưu ý:**

* Type Selector **không** chọn các phần tử có thuộc tính id hoặc class cụ thể.
* Mặc dù tiện lợi, việc sử dụng quá nhiều Type Selector có thể dẫn đến CSS kém linh hoạt và khó maintain.
* Nên kết hợp Type Selector với các Selector khác như Class Selector hoặc Descendant Selector để tăng tính chính xác và linh hoạt cho CSS của bạn.

### 3.4.3. Identity selector

Trong CSS, **Identity Selector** không thực sự tồn tại như một khái niệm riêng biệt. Có thể bạn đang nhầm lẫn với một trong những Selector khác thường được sử dụng để chọn các phần tử cụ thể:

**1. ID Selector:**

Nếu bạn đang nghĩ đến một cách chọn một phần tử duy nhất trên trang web, thì đó có thể là **ID Selector**. Nó được biểu thị bằng **dấu thăng (#)**, tiếp theo là ID duy nhất của phần tử HTML. Ví dụ:

Trong HTML, mỗi phần tử chỉ có thể có một ID duy nhất, đảm bảo Selector này nhắm mục tiêu chính xác một phần tử trên toàn bộ trang.

**2. Class Selector:**

Mặc dù không phải là chọn duy nhất, **Class Selector** có thể nhắm mục tiêu các phần tử chia sẻ cùng một class. Nó được biểu thị bằng dấu chấm (.), tiếp theo là tên của class.

Nhiều phần tử có thể có cùng một class, cho phép bạn định dạng một nhóm các phần tử liên quan.

**3. Descendant Selector:**

Nếu bạn đang nghĩ đến việc chọn một phần tử dựa trên vị trí của nó trong cấu trúc HTML, thì đó có thể là **Descendant Selector**. Nó cho phép bạn chọn các phần tử là con, cháu, v.v. của một phần tử khác. Ví dụ:

Điều này chọn tất cả các tiêu đề h2 nằm bên trong các phần tử div.

**Không có Selector nào trong CSS được gọi chính xác là "Identity Selector."** Tuy nhiên, tùy thuộc vào nhu cầu chọn của bạn, một trong những Selector trên có thể phù hợp hơn cả. Nếu bạn cung cấp thêm thông tin về những gì bạn muốn đạt được, tôi có thể hướng dẫn bạn chi tiết hơn.

### 3.4.4. Class selector

Trong CSS, **Class Selector** là một công cụ mạnh mẽ cho phép bạn chọn các phần tử HTML có thuộc tính class cụ thể. Nó giúp bạn định dạng nhiều phần tử cùng một lúc theo cách nhất quán và linh hoạt.

**Cách sử dụng Class Selector:**

1. **Thêm thuộc tính** class **vào phần tử HTML:** Đầu tiên, bạn cần thêm thuộc tính class vào các phần tử HTML mà bạn muốn định dạng. Thuộc tính này được đặt trong thẻ mở của phần tử, với cú pháp class="class-name". Ví dụ:

content\_copy

1. **Sử dụng Class Selector trong CSS:** Tiếp theo, trong bộ quy tắc CSS của bạn, hãy viết dấu chấm (.) tiếp theo là tên của class mà bạn muốn chọn. Ví dụ:

Đây sẽ áp dụng các kiểu tương ứng cho tất cả các phần tử có class important-text và primary-button.

**Lợi ích của Class Selector:**

* **Tính linh hoạt:** Bạn có thể sử dụng cùng một class cho nhiều phần tử khác nhau trên trang web của bạn, giúp tiết kiệm thời gian và duy trì tính nhất quán.
* **Kiểm soát chính xác:** Bạn có thể tạo các class khác nhau để định dạng các nhóm phần tử khác nhau, cho phép bạn tùy chỉnh giao diện của trang web một cách chi tiết.
* **Dễ dàng bảo trì:** Khi bạn cần thay đổi kiểu cho một nhóm phần tử, bạn chỉ cần cập nhật bộ quy tắc CSS của class đó, thay vì phải tìm kiếm và thay đổi từng phần tử trong HTML.

### 3.4.5. Descendant selector

Trong CSS, **Descendant Selector** cho phép bạn chọn các phần tử HTML là con cháu của một phần tử khác trong cấu trúc HTML. Nó rất hữu ích khi bạn muốn định dạng các phần tử có mối quan hệ phân cấp rõ ràng với nhau.

**Cú pháp:**

Cú pháp của Descendant Selector rất đơn giản. Bạn chỉ cần viết tên của phần tử cha, tiếp theo là một khoảng trắng và sau đó là tên của phần tử con mà bạn muốn chọn. Ví dụ:

Trong ví dụ này, Selector sẽ chọn tất cả các tiêu đề h2 nằm bên trong bất kỳ phần tử div nào trên trang.

**Các khái niệm liên quan:**

* **Cha (Parent):** Phần tử mà bạn sử dụng trong Selector là phần tử cha của các phần tử mà bạn muốn chọn.
* **Con (Child):** Phần tử trực tiếp nằm bên trong phần tử cha là con của phần tử cha.
* **Cháu (Descendant):** Bất kỳ phần tử nào nằm bên trong, bất kể bao nhiêu cấp sâu, của phần tử cha đều được coi là cháu.

Lưu ý:

* Descendant Selector có thể đi sâu nhiều cấp trong cấu trúc HTML. Ví dụ, nav ul li a sẽ chọn tất cả các liên kết nằm bên trong các danh sách không thứ tự (ul) nằm bên trong các phần tử nav.
* Sử dụng Descendant Selector quá nhiều có thể khiến CSS của bạn trở nên khó đọc và khó maintain. Nên cân nhắc kết hợp nó với các Selector khác như Class Selector hoặc Type Selector để tăng tính chính xác và linh hoạt.

### 3.4.6. Child selector Trong CSS, **Child Selector** cho phép bạn chọn các phần tử HTML là **con trực tiếp** của một phần tử khác. Nó khác với Descendant Selector ở chỗ chỉ chọn con trực tiếp, không bao gồm các cháu hoặc hậu duệ khác trong cấu trúc HTML.

**-Cú pháp:**

Cú pháp của Child Selector rất đơn giản, sử dụng dấu lớn hơn (>) ngay sau tên của phần tử cha, tiếp theo là tên của phần tử con mà bạn muốn chọn. Ví dụ:

Trong ví dụ này, Selector sẽ chọn tất cả các đoạn văn là con trực tiếp của bất kỳ phần tử div nào trên trang, nhưng không bao gồm các đoạn văn nằm bên trong các phần tử khác bên trong div.

Lưu ý:

* Child Selector chỉ chọn các con trực tiếp, không bao gồm các cháu hoặc hậu duệ khác.
* Sử dụng Child Selector có thể giúp bạn nhắm mục tiêu các phần tử trong một cấp cụ thể của cấu trúc HTML, mang lại nhiều kiểm soát hơn so với Type Selector hoặc Descendant Selector.
* Tương tự như các Selector khác, sử dụng Child Selector quá nhiều có thể khiến CSS của bạn trở nên phức tạp. Cân nhắc kết hợp nó với các Selector khác để đạt được độ chính xác và hiệu quả mong muốn.

### 3.4.7. Adjacent sibling selector

Trong CSS, Adjacent Sibling Selector cho phép bạn chọn một phần tử chỉ khi nó là anh chị em ngay sau một phần tử khác cụ thể. Điều này có nghĩa là hai phần tử phải có cùng cha mẹ trong cấu trúc HTML và nằm cạnh nhau, không có phần tử nào khác ở giữa.

Cú pháp:

Cú pháp của Adjacent Sibling Selector sử dụng dấu cộng (+) ngay sau tên của phần tử đầu tiên, tiếp theo là tên của phần tử thứ hai mà bạn muốn chọn.

Lưu ý:

* Chỉ phần tử thứ hai được chọn, không phải phần tử đầu tiên.
* Hai phần tử phải có cùng cha mẹ trong cấu trúc HTML.
* Không có phần tử nào khác được phép nằm giữa hai phần tử được chọn.

### 3.4.8. Attribute selector

Trong CSS, **Attribute Selector** cho phép bạn chọn các phần tử HTML dựa trên **tên, giá trị hoặc sự có mặt của thuộc tính của chúng**. Điều này rất hữu ích khi bạn muốn định dạng các phần tử có các đặc điểm cụ thể trong HTML.

**Cú pháp:**

Cú pháp của Attribute Selector sử dụng ngoặc vuông [] để bao quanh tên của thuộc tính, theo sau là các toán tử so sánh tùy chọn và giá trị của thuộc tính. Dưới đây là một số cách sử dụng phổ biến:

* **Chọn các phần tử có thuộc tính nhất định:**
* **Chọn các phần tử có thuộc tính với giá trị cụ thể:**
* **Chọn các phần tử có thuộc tính bắt đầu bằng giá trị nhất định:**
* **Chọn các phần tử có thuộc tính chứa giá trị nhất định:**
* **Chọn các phần tử có thuộc tính kết thúc bằng giá trị nhất định:**

**Ví dụ sử dụng:**

* Thêm đường viền cho tất cả các hình ảnh có thuộc tính alt:
* Thay đổi màu nền cho tất cả các liên kết có thuộc tính href bắt đầu bằng http:
* Tăng kích thước font cho tất cả các thẻ input có thuộc tính type là text:

**Lưu ý:**

* Bạn có thể kết hợp Attribute Selector với các Selector khác để nhắm mục tiêu chính xác hơn.
* Sử dụng quá nhiều Attribute Selector có thể khiến CSS của bạn trở nên khó đọc và khó maintain.
* Cân nhắc sử dụng Class Selector khi có thể, vì nó thường linh hoạt hơn và dễ maintain hơn.

### 3.4.9. Pseudo class selector

Trong CSS, Pseudo-class Selector là một cách linh hoạt để chọn các phần tử HTML dựa trên trạng thái của chúng, chứ không phải dựa trên loại thẻ, ID, class hoặc thuộc tính của chúng. Đây là một công cụ mạnh mẽ cho phép bạn thêm các hiệu ứng động và tương tác cho trang web của mình.

Cú pháp:

Pseudo-class Selector được viết bằng dấu hai điểm (:) ngay sau tên của phần tử hoặc Selector khác, tiếp theo là tên của Pseudo-class. Dưới đây là một số Pseudo-class phổ biến:

* :hover: Chọn phần tử khi người dùng di chuột qua nó.
* :active: Chọn phần tử khi người dùng đang nhấn giữ nó.
* :focus: Chọn phần tử khi nó đang có tiêu điểm, thường khi người dùng đang nhập văn bản vào nó.
* :link: Chọn liên kết chưa được truy cập.
* :visited: Chọn liên kết đã được truy cập.
* :checked: Chọn các hộp kiểm hoặc nút radio đã được chọn.
* :disabled: Chọn các phần tử bị vô hiệu hóa.

Lưu ý:

* Bạn có thể kết hợp Pseudo-class Selector với các Selector khác để nhắm mục tiêu chính xác hơn.
* Sử dụng quá nhiều Pseudo-class Selector có thể khiến CSS của bạn trở nên phức tạp và khó maintain.
* Cân nhắc sử dụng các Pseudo-class phù hợp với hành vi và tương tác của người dùng trên trang web của bạn.

### 3.4.10. Group selector

Trong CSS, Group Selector không thực sự là một thuật ngữ chính thức hay có một cú pháp riêng biệt. Nó thường được dùng để mô tả phương pháp ghép nhóm nhiều Selector lại với nhau bằng dấu "," để áp dụng cùng một bộ quy tắc CSS cho chúng.

Cách sử dụng Group Selector:

1. Viết các Selector bạn muốn nhóm lại: Bắt đầu bằng bất kỳ loại Selector hợp lệ nào như Class Selector, ID Selector, Type Selector, v.v...
2. Phân cách các Selector bằng dấu ",": Viết dấu "," giữa mỗi Selector để tạo thành một danh sách các phần tử mục tiêu.
3. Áp dụng các kiểu CSS: Sau danh sách Selector, đặt ngoặc nhọn "{" và viết các quy tắc CSS bên trong chúng, tương tự như bất kỳ quy tắc CSS thông thường nào khác.

## 3.5. Đơn vị đo lường CSS

CSS sử dụng nhiều loại đơn vị đo lường khác nhau để xác định kích thước, vị trí, v.v. của các phần tử HTML. Việc lựa chọn đơn vị phù hợp là rất quan trọng để đảm bảo bố cục trang web hiển thị chính xác và đồng nhất trên các thiết bị khác nhau.

Dưới đây là một số loại đơn vị đo lường phổ biến trong CSS:

1. Đơn vị tuyệt đối:

* px (pixel): Đơn vị cơ bản để đo kích thước trên màn hình. 1px tương ứng với một điểm ảnh trên màn hình.
* pt (point): Bằng 1/72 inch (khoảng 0.35mm). Thường được sử dụng để xác định kích thước phông chữ.
* cm (centimeter): Đơn vị đo lường chiều dài theo hệ mét.
* mm (millimeter): 1/10 cm.
* in (inch): Đơn vị đo lường chiều dài theo hệ inch.
* pc (pica): Bằng 12pt (khoảng 16mm).

2. Đơn vị tương đối:

* em: Kích thước tương đối so với kích thước phông chữ của phần tử cha. Ví dụ, 1em bằng 16px nếu kích thước phông chữ của phần tử cha là 16px.
* rem: Kích thước tương đối so với kích thước phông chữ của phần tử gốc (html). 1rem bằng 16px nếu kích thước phông chữ của phần tử gốc là 16px.
* % (phần trăm): Kích thước tương đối so với kích thước của phần tử cha. Ví dụ, 50% bằng một nửa kích thước của phần tử cha.

3. Đơn vị khác:

* vh (viewport height): 1% chiều cao của khung nhìn (viewport).
* vw (viewport width): 1% chiều rộng của khung nhìn.
* vmin: Giá trị nhỏ nhất giữa vw và vh.
* vmax: Giá trị lớn nhất giữa vw và vh.

Lựa chọn đơn vị phù hợp:

* Kích thước cố định: Sử dụng đơn vị tuyệt đối như px, pt, cm, mm, in cho các yếu tố có kích thước cố định, không thay đổi theo kích thước màn hình.
* Kích thước linh hoạt: Sử dụng đơn vị tương đối như em, rem, % cho các yếu tố có kích thước thay đổi theo kích thước màn hình.
* Vị trí: Sử dụng đơn vị như px, em, % cho vị trí của các yếu tố.
* Đơn vị viewport: Sử dụng đơn vị vh, vw, vmin, vmax cho các yếu tố có kích thước tương ứng với khung nhìn.

Lưu ý:

* Nên sử dụng đơn vị phù hợp nhất cho từng trường hợp cụ thể.
* Tránh sử dụng quá nhiều đơn vị khác nhau trong cùng một trang web.
* Sử dụng các công cụ hỗ trợ như CSS Lint để kiểm tra lỗi trong mã CSS.

## 3.6. Kế thừa thuộc tính

Kế thừa thuộc tính là một tính năng quan trọng trong CSS cho phép các thuộc tính được truyền từ phần tử cha sang phần tử con. Nhờ vậy, bạn có thể định kiểu cho nhiều phần tử cùng lúc mà không cần phải lặp lại các thuộc tính cho từng phần tử.

Cách thức hoạt động:

* Các thuộc tính CSS có thể được kế thừa từ phần tử cha sang phần tử con.
* Một số thuộc tính không thể kế thừa, ví dụ như position, display, float, clear, visibility, cursor, outline, z-index.
* Khi một phần tử con không có định nghĩa cho một thuộc tính nào đó, nó sẽ kế thừa giá trị của thuộc tính đó từ phần tử cha.
* Nếu phần tử cha cũng không có định nghĩa cho thuộc tính đó, nó sẽ tiếp tục được kế thừa từ phần tử cha của nó, cho đến khi tìm thấy phần tử có định nghĩa cho thuộc tính đó.

Lợi ích:

* Giúp giảm thiểu lượng mã CSS cần viết.
* Giúp cho việc định kiểu trang web trở nên dễ dàng và hiệu quả hơn.
* Giúp đảm bảo tính đồng nhất trong việc định kiểu trang web.

Lưu ý:

* Thứ tự ưu tiên của kế thừa thuộc tính:
  1. Thuộc tính được định nghĩa trong phần tử con
  2. Thuộc tính được định nghĩa trong phần tử cha
  3. Thuộc tính được định nghĩa trong các phần tử cha tiếp theo
  4. Thuộc tính được định nghĩa trong trình duyệt
* Có thể sử dụng thuộc tính inherit để buộc phần tử con kế thừa giá trị của thuộc tính từ phần tử cha, ngay cả khi phần tử con đã có định nghĩa cho thuộc tính đó.

## 3.7. Các nhóm thuộc tính trong CSS

- Thuộc tính định dạng font chữ, văn bản (font, text)

* font-family: Xác định họ phông chữ của phần tử.
* font-size: Xác định kích thước phông chữ của phần tử.
* font-weight: Xác định độ đậm của phông chữ.
* font-style: Xác định kiểu chữ (in nghiêng, bình thường, đậm).
* text-align: Xác định cách căn chỉnh văn bản trong phần tử.
* text-decoration: Xác định kiểu trang trí cho văn bản (underline, overline, strikethrough).
* border: Xác định kiểu, độ rộng và màu sắc của viền border.
* box-shadow: Tạo hiệu ứng bóng đổ cho phần tử.

- Thuộc tính định dạng nền (background)

**1.** background-color**:** Xác định màu nền cho phần tử.

**2.** background-image**:** Xác định hình ảnh nền cho phần tử.

**3.** background-position**:** Xác định vị trí của hình ảnh nền.

**4.** background-repeat**:** Xác định cách thức lặp lại hình ảnh nền.

**5.** background-size**:** Xác định kích thước của hình ảnh nền.

**6.** background-attachment**:** Xác định cách thức di chuyển hình ảnh nền khi cuộn trang.

- Mô hình hộp (box model)

Mô hình hộp (box model) là một khái niệm quan trọng trong CSS, dùng để mô tả cấu trúc của một phần tử HTML. Nó bao gồm các thành phần sau:

1. Nội dung (content): Là phần nội dung chính của phần tử, như văn bản, hình ảnh, video, v.v.

2. Padding: Là khoảng cách giữa nội dung và viền của phần tử.

3. Viền (border): Là đường viền bao quanh phần tử.

4. Lề (margin): Là khoảng cách giữa phần tử và các phần tử xung quanh.

- Các thuộc tính định margin, padding, border

1. Margin:

* margin-top: Định vị khoảng cách từ phần tử đến cạnh trên của phần tử cha.
* margin-right: Định vị khoảng cách từ phần tử đến cạnh phải của phần tử cha.
* margin-bottom: Định vị khoảng cách từ phần tử đến cạnh dưới của phần tử cha.
* margin-left: Định vị khoảng cách từ phần tử đến cạnh trái của phần tử cha.
* margin: Thuộc tính rút gọn để định vị đồng thời cả bốn cạnh margin.

2. Padding:

* padding-top: Định vị khoảng cách từ nội dung đến cạnh trên của phần tử.
* padding-right: Định vị khoảng cách từ nội dung đến cạnh phải của phần tử.
* padding-bottom: Định vị khoảng cách từ nội dung đến cạnh dưới của phần tử.
* padding-left: Định vị khoảng cách từ nội dung đến cạnh trái của phần tử.
* padding: Thuộc tính rút gọn để định vị đồng thời cả bốn cạnh padding.

3. Border:

* border-width: Định vị độ rộng của viền.
* border-style: Định vị kiểu viền (solid, dotted, dashed, double, none).
* border-color: Định vị màu sắc của viền.
* border-top: Thuộc tính rút gọn để định vị viền cho cạnh trên.
* border-right: Thuộc tính rút gọn để định vị viền cho cạnh phải.
* border-bottom: Thuộc tính rút gọn để định vị viền cho cạnh dưới.
* border-left: Thuộc tính rút gọn để định vị viền cho cạnh trái.
* border: Thuộc tính rút gọn để định vị đồng thời viền cho cả bốn cạnh.

- Thuộc tính định dạng cách hiển thị (display)

Dưới đây là một số giá trị phổ biến của thuộc tính display:

* block**:** Hiển thị phần tử dưới dạng khối, chiếm toàn bộ chiều rộng của phần tử cha và bắt đầu một dòng mới.
* inline**:** Hiển thị phần tử dưới dạng nội tuyến, nằm trên cùng dòng với các phần tử khác.
* inline-block**:** Kết hợp đặc điểm của block và inline, cho phép phần tử hiển thị như một khối nhưng vẫn nằm trên cùng dòng với các phần tử khác.
* none**:** Ẩn phần tử khỏi trang web.
* flex**:** Hiển thị phần tử theo bố cục flexbox, cho phép bạn bố trí các phần tử con một cách linh hoạt.
* grid**:** Hiển thị phần tử theo bố cục grid, cho phép bạn tạo bố cục dạng lưới với các cột và hàng.

- Thuộc tính xác định vị trí (position)

* static**:** Vị trí mặc định của phần tử, tuân theo bố cục tự nhiên của trang web.
* relative**:** Vị trí tương đối so với vị trí ban đầu của phần tử. Di chuyển phần tử bằng các thuộc tính top, bottom, left, right.
* absolute**:** Vị trí tuyệt đối so với phần tử cha gần nhất có vị trí relative, absolute hoặc fixed.
* fixed**:** Vị trí cố định so với khung nhìn của trình duyệt, không di chuyển khi cuộn trang.
* sticky**:** Vị trí kết hợp giữa relative và fixed. Phần tử sẽ di chuyển theo vị trí relative cho đến khi đến vị trí được xác định,

## 3.8. Float & Clear

Float:

* Thuộc tính float cho phép bạn di chuyển một phần tử sang trái (left) hoặc phải (right) của phần tử cha, cho các nội dung khác "chảy" xung quanh nó.
* Phần tử được thiết lập float sẽ tách khỏi dòng thông thường và tự định vị theo vị trí được chỉ định.
* Thường được sử dụng để tạo bố cục hai cột cạnh nhau, sidebar, hình ảnh nổi, v.v.

Clear:

* Thuộc tính clear giúp loại bỏ các phần tử khác "chảy" xung quanh một phần tử được thiết lập float.
* Các giá trị phổ biến:
  + none: Không xóa.
  + left: Xóa các phần tử nổi bên trái.
  + right: Xóa các phần tử nổi bên phải.
  + both: Xóa cả các phần tử nổi bên trái và phải.
* Thường được sử dụng sau phần tử cuối cùng được float trong một nhóm để đảm bảo các nội dung tiếp theo hiển thị bên dưới đúng vị trí.

## 3.9. Flex

Flexbox ( viết tắt của Flexible Box) là một mô hình bố cục một chiều mạnh mẽ và linh hoạt trong CSS, cho phép bạn dễ dàng sắp xếp các phần tử con theo chiều ngang hoặc chiều dọc. Nó cung cấp nhiều tính năng giúp bạn tạo bố cục đáp ứng (responsive) và phức tạp dễ dàng hơn so với phương pháp float và clear truyền thống.

Các khái niệm chính của Flexbox:

* Flex Container: Phần tử cha chứa các phần tử con được sắp xếp bằng Flexbox.
* Flex Item: Các phần tử con bên trong Flex container.
* Flex Direction: Xác định hướng chính của bố cục (row - ngang, column - dọc, row-reverse, column-reverse).
* Justify Content: Xác định cách phân bố các Flex item dọc theo trục chính của Flex container (flex-start, flex-end, center, space-between, space-around).
* Align Items: Xác định cách căn chỉnh các Flex item theo trục phụ của Flex container (flex-start, flex-end, center, stretch).
* Flex Wrap: Xác định cách xử lý khi các Flex item không vừa trong một dòng (nowrap - không bọc dòng, wrap - bọc dòng, wrap-reverse - bọc dòng ngược).

Ưu điểm của Flexbox:

* Dễ dàng tạo bố cục đáp ứng: Flexbox giúp bạn dễ dàng điều chỉnh bố cục của các phần tử trên các thiết bị khác nhau.
* Linh hoạt cao: Cung cấp nhiều tùy chọn để sắp xếp và căn chỉnh các phần tử.
* Giảm thiểu mã CSS: Loại bỏ sự cần sử dụng float và clear,

## 3.10. Grid

Grid Layout là một mô hình bố cục hai chiều mạnh mẽ trong CSS, cho phép bạn sắp xếp các phần tử con thành các hàng và cột giống như một bảng lưới. Nó cung cấp nhiều tính năng linh hoạt giúp bạn tạo bố cục phức tạp, đáp ứng và dễ quản lý hơn so với các phương pháp bố cục truyền thống.

**Các khái niệm chính của Grid Layout:**

* **Grid Container:** Phần tử cha chứa các phần tử con được sắp xếp bằng Grid Layout.
* **Grid Item:** Các phần tử con bên trong Grid container.
* **Grid Track:** Đường phân chia hàng và cột trong Grid container.
* **Grid Line:** Giao điểm của đường hàng và cột, xác định vị trí của Grid item.
* **Grid Area:** Vùng được tạo bởi sự kết hợp của các đường hàng và cột, dùng để định vị chính xác Grid item.
* **Grid Auto-placement:** Tự động sắp xếp các Grid item vào các ô trống trong Grid container.

**Ưu điểm của Grid Layout:**

* **Bố cục linh hoạt:** Cho phép bạn dễ dàng tạo bố cục phức tạp và đáp ứng trên các thiết bị khác nhau.
* **Kiểm soát chính xác:** Cung cấp nhiều tùy chọn để định vị và sắp xếp các Grid item.
* **Cấu trúc rõ ràng:** Giúp code CSS dễ hiểu và dễ quản lý hơn.

## 3.11. CSS3

- CSS3 là gì?

CSS3 là viết tắt của Cascading Style Sheets level 3, là phiên bản nâng cao của CSS (Cascading Style Sheets). Nó được sử dụng để cấu trúc, tạo kiểu và định dạng các trang web, mang đến nhiều tính năng mới và mạnh mẽ hơn so với CSS2.

Một số điểm nổi bật của CSS3:

* Bổ sung nhiều tính năng mới: CSS3 mang đến nhiều tính năng mới giúp bạn tạo giao diện web đẹp mắt và ấn tượng hơn, ví dụ như:
  + Bo tròn góc
  + Tạo độ bóng
  + Gradient (chuyển màu)
  + Animations (hiệu ứng chuyển động)
  + Transitions (chuyển đổi mượt mà)
  + Bố cục mới (multiple-column, flexbox, grid layout)
* Khả năng tương thích cao: CSS3 được hỗ trợ bởi hầu hết các trình duyệt web hiện đại, giúp đảm bảo trang web của bạn hiển thị đẹp mắt trên mọi thiết bị.
* Dễ dàng sử dụng: CSS3 có cú pháp tương tự như CSS2, do đó bạn có thể dễ dàng học hỏi và sử dụng.

Lợi ích khi sử dụng CSS3:

* Tạo giao diện web đẹp mắt và ấn tượng: CSS3 cung cấp nhiều tính năng mới giúp bạn tạo giao diện web đẹp mắt và thu hút người dùng.
* Tăng khả năng tương thích: CSS3 được hỗ trợ bởi hầu hết các trình duyệt web hiện đại, giúp đảm bảo trang web của bạn hiển thị đẹp mắt trên mọi thiết bị.
* Giảm thời gian phát triển: CSS3 giúp bạn tiết kiệm thời gian phát triển trang web nhờ các tính năng mới như flexbox và grid layout.
* Tăng hiệu suất trang web: CSS3 giúp tối ưu hóa hiệu suất trang web bằng cách giảm thiểu dung lượng mã code.

- CSS3 Selectors

Trong CSS3, các selector (bộ chọn) đóng vai trò quan trọng, cho phép bạn xác định chính xác các thành phần HTML cần áp dụng kiểu dáng. So với CSS2, CSS3 mang đến nhiều loại selector mới đa dạng và mạnh mẽ hơn, giúp bạn linh hoạt hơn trong việc tạo kiểu cho trang web.

* + Các loại selector phổ biến trong CSS3
* **Simple selector:** Chọn dựa trên tên thẻ (ví dụ: p, h1), ID (#unique-id), class (.my-class)
* **Descendant combinator:** Chọn thành phần con (div > p chọn paragraph con của div)
* **Child combinator:** Chọn thành phần con trực tiếp (div > p chọn paragraph là con trực tiếp của div)
* **Universal selector:** Chọn tất cả các thành phần (\*)
* **Attribute selector:** Chọn dựa trên thuộc tính của thành phần ([href="link.html"] chọn thành phần có thuộc tính href là "link.html")
* **Pseudo-class:** Chọn dựa trên trạng thái của thành phần (:hover chọn khi di chuột qua, :active chọn khi nhấn giữ)
* **Pseudo-element:** Chọn các phần cụ thể của thành phần (:first-child chọn con đầu tiên, ::after chọn nội dung sau thành phần)

+ Ưu điểm

* **Độ chính xác cao:** Chọn đúng các thành phần cần thiết, tránh ảnh hưởng đến thành phần khác.
* **Mạnh mẽ và linh hoạt:** Cho phép tạo kiểu phức tạp với nhiều điều kiện kết hợp.
* **Cải thiện hiệu suất:** Selector chính xác giúp trình duyệt render trang nhanh hơn.
* **Tạo code sạch và dễ bảo trì:**

- CSS3 Rounded Corners

Trong CSS3, tính năng bo tròn góc (border-radius) là một công cụ mạnh mẽ giúp bạn tạo giao diện web mềm mại, thân thiện và hiện đại hơn. Bằng cách bo tròn các góc của các thành phần HTML như hình ảnh, button, hộp thoại, bạn có thể làm tăng tính thẩm mỹ và thu hút người dùng.

+Sử dụng thuộc tính border-radius

Thuộc tính border-radius cho phép bạn xác định bán kính của đường bo tròn cho từng góc của thành phần. Bạn có thể sử dụng:

* Một giá trị duy nhất: Điều này sẽ tạo đường bo tròn đồng đều cho tất cả các góc.
* Hai giá trị ngăn cách bằng dấu cách: Xác định bán kính cho góc trên trái và dưới phải đồng thời.
* Ba giá trị: Xác định bán kính cho góc trên trái, trên phải và dưới phải.
* Bốn giá trị: Xác định bán kính riêng biệt cho mỗi góc (trên trái, trên phải, dưới phải, dưới trái).

- CSS3 Border Images

**CSS3 Border Images** là một tính năng mạnh mẽ cho phép bạn sử dụng hình ảnh làm khung viền thay vì đường viền đơn giản. Điều này mở ra nhiều khả năng sáng tạo, giúp bạn thiết kế website độc đáo và bắt mắt hơn.

**Nguyên lý hoạt động:**

1. **Hình ảnh được chia thành 9 phần:** Giống như trò chơi ô ăn quan, hình ảnh được chia thành 4 góc, 4 cạnh và 1 phần trung tâm.
2. **Góc được đặt đúng vị trí:** Góc của hình ảnh được đặt chính xác tại các góc của thành phần HTML.
3. **Cạnh được lặp lại hoặc kéo dài:** Các cạnh được lặp lại theo chiều ngang/dọc hoặc kéo dài tùy thuộc thuộc tính bạn đặt.
4. **Phần trung tâm tùy chỉnh:** Phần trung tâm có thể lặp lại hoặc kéo dài tùy ý.

**Cấu trúc code:**

* Sử dụng thuộc tính border-image với 5 thuộc tính phụ:
  + border-image-source: Đường dẫn đến hình ảnh.
  + border-image-slice: Xác định cách chia 9 phần của hình ảnh.
  + border-image-width: Chiều rộng đường viền (dùng hình ảnh).
  + border-image-outset: Mở rộng khung viền ra ngoài.
  + border-image-repeat: Xác định cách lặp lại các cạnh và phần trung tâm.

- CSS3 Backgrounds

Trong CSS3, tính năng nền (background) được nâng cấp đáng kể, cung cấp nhiều công cụ giúp bạn tạo ra những nền trang web bắt mắt, linh hoạt và hiệu ứng phong phú. Hãy cùng khám phá một số tính năng nổi bật:

**1. Hình nền (background-image):**

* Không giới hạn một hình: Sử dụng nhiều hình ảnh chồng lên nhau với background-position để xác định vị trí từng hình.
* Điều khiển kích thước: Kiểm soát kích thước và tỉ lệ hình nền với background-size.
* Lặp lại linh hoạt: Chọn cách lặp lại hình nền theo chiều ngang, dọc hoặc cả hai với background-repeat.
* Gradient đầy màu sắc: Tạo nền chuyển màu mượt mà với background-image: linear-gradient() hoặc background-image: radial-gradient().

**2. Màu nền (background-color):**

* Kết hợp màu và hình ảnh: Sử dụng background-color làm lớp nền cho hình ảnh trong suốt.
* Màu RGBA và độ mờ: Sử dụng kênh alpha trong RGBA để tạo màu nền bán trong suốt với background-color: rgba(r, g, b, a).

**3. Điểm nhấn cho nội dung (background-attachment):**

* Cuộn cùng nội dung: Giữ hình nền cố định khi cuộn trang với background-attachment: fixed.
* Cuộn theo nội dung: Để hình nền cuộn cùng nội dung trang với background-attachment: scroll.

**4. Tạo đường viền với nền (background-clip):**

* Nền trong nội dung: Giới hạn nền chỉ áp dụng trong nội dung với background-clip: padding-box.
* Nền bao gồm cả đường viền: Áp dụng nền cho cả nội dung và đường viền với background-clip: border-box.

**5. Tùy chỉnh hình nền theo kích thước thiết bị (media queries):**

* Hiển thị hình nền khác nhau trên màn hình lớn và nhỏ.

**Ưu điểm:**

* Tạo giao diện phong phú, thu hút thị giác.
* Kiểm soát chính xác hiệu ứng nền.
* Cải thiện trải nghiệm người dùng.

- CSS Gradients

Trong CSS3, gradients (chuyển màu) là một tính năng mạnh mẽ cho phép bạn tạo ra chuyển đổi màu sắc mượt mà và đẹp mắt trên nền trang web, nút bấm, hộp thoại, v.v... Nó không chỉ tăng thêm tính thẩm mỹ mà còn tạo chiều sâu và hiệu ứng thị giác ấn tượng.

Hai loại gradient chính:

* Linear Gradients: Chuyển màu theo một hướng xác định (ngang, dọc, chéo).
* Radial Gradients: Chuyển màu từ một điểm trung tâm ra ngoài theo hình tròn hoặc elip.

Cấu trúc code:

Sử dụng thuộc tính background-image với giá trị là linear-gradient() hoặc radial-gradient(). Bên trong ngoặc đơn, bạn khai báo các màu sắc muốn chuyển đổi, vị trí điểm bắt đầu và kết thúc,

- CSS3 Shadows

Trong CSS3, tính năng box-shadow cho phép bạn thêm bóng đổ cho các thành phần HTML, tạo chiều sâu và hiệu ứng 3D tinh tế. Bóng đổ không chỉ giúp website trông đẹp mắt hơn mà còn làm nổi bật các thành phần quan trọng, thu hút sự chú ý của người dùng.

* Sử dụng thuộc tính box-shadow với nhiều giá trị ngăn cách bằng dấu phẩy, mỗi giá trị gồm:
  + **Offset X và Y:** Xác định vị trí bóng đổ so với cạnh bên trái và trên cùng của thành phần (dùng pixel, phần trăm hoặc em).
  + **Blur:** Điều chỉnh độ mờ của bóng đổ (dùng pixel).
  + **Spread:** Mở rộng kích thước bóng đổ ra ngoài (dùng pixel).
  + **Màu sắc:** Xác định màu sắc của bóng đổ (dùng tên màu hoặc mã màu hex).
* Bạn có thể thêm nhiều bóng đổ bằng cách ngăn cách các giá trị bằng dấu phẩy.

- CSS3 Fonts

Trong CSS3, tính năng Fonts được mở rộng đáng kể, không chỉ giới hạn trong phông chữ mặc định của hệ thống mà cho phép bạn sử dụng phông chữ tùy chỉnh, tạo nên giao diện web độc đáo và mang đậm dấu ấn riêng.

Sử dụng phông chữ tùy ý với font-face

* Khai báo thuộc tính @font-face trong stylesheet của bạn.
* Xác định tên phông chữ, đường dẫn đến file phông (định dạng WOFF, WOFF2, OTF, TTF), định dạng phông và các tùy chọn khác (kiểu dáng, độ dày).

- CSS3 2D Transforms

Trong CSS3, tính năng transform cho phép bạn thực hiện các phép biến đổi 2 chiều (dọc và ngang) cho các thành phần HTML. Bằng cách này, bạn có thể xoay, thu phóng, dịch chuyển và nghiêng chúng, tạo ra hiệu ứng chuyển động mượt mà và ấn tượng.

Các phép biến đổi chính

* **Translate:** Dịch chuyển thành phần theo trục X và Y.
* **Rotate:** Xoay thành phần theo một góc xác định.
* **Scale:** Thay đổi kích thước của thành phần.
* **Skew:** Nghiêng thành phần theo trục X và Y.

- CSS3 3D Transforms

Trong thế giới web phẳng, CSS3 3D Transforms mở ra cánh cửa bước vào không gian ba chiều, cho phép bạn thực hiện các phép biến đổi xoay, dịch chuyển và thu phóng trên trục X, Y và Z cho các thành phần HTML. Điều này giúp bạn tạo ra hiệu ứng 3D chân thực, đẹp mắt và thu hút người dùng.

So với 2D Transfoms

* Thêm trục Z, tạo chiều sâu cho hiệu ứng.
* Sử dụng các hàm xoay 3D chuyên biệt như rotateX(), rotateY(), rotateZ().
* Cần thiết lập perspective để mô phỏng điểm nhìn 3D.

+Cấu trúc

1. **Perspective:** Xác định điểm nhìn 3D với perspective(đơn vị).
2. Biến đổi 3D: Áp dụng các phép biến đổi trên trục X, Y, Z với:

* rotateX(góc)
* rotateY(góc)
* rotateZ(góc)
* translateX(đơn vị)
* translateY(đơn vị)
* translateZ(đơn vị)

- CSS3 Transitions

Trong CSS3, tính năng transitions mang đến sức mạnh biến hóa cho giao diện web. Nó cho phép bạn tạo hiệu ứng chuyển động mượt mà và đẹp mắt khi các thuộc tính CSS của thành phần thay đổi, chẳng hạn như khi di chuột qua, click, cuộn trang, v.v...

+Bản chất

Transitions hoạt động bằng cách xác định:

* **Thuộc tính CSS:** Thuộc tính nào sẽ được animated (ví dụ: background-color, opacity, margin).
* **Thời gian:** Khoảng thời gian diễn ra hiệu ứng chuyển động (tính bằng giây hoặc mili giây).
* **Easing function:** Kiểu chuyển động của hiệu ứng (linear, ease-in, ease-out, ease-in-out,

- CSS3 Multiple Columns

Trong CSS3, tính năng multicolumn cho phép bạn chia các thành phần HTML thành nhiều cột, giúp bạn tạo bố cục trang web linh hoạt, dễ đọc và phù hợp với nhiều kích thước màn hình. Điều này đặc biệt hữu dụng cho các nội dung dạng văn bản, danh sách, sản phẩm, v.v...

+Nguyên lí hoạt động

* Xác định số cột mong muốn với thuộc tính column-count.
* Kiểm soát khoảng cách giữa các cột với column-gap.
* Tùy chỉnh đường viền giữa các cột với column-rule.
* Điều khiển cách thức ngắt dòng nội dung giữa các cột với break-before và break-after.

- CSS3 Animations

Trong CSS3, tính năng **Animations** mở ra cánh cửa cho thế giới chuyển động đầy màu sắc và ấn tượng trên website. Nó cho phép bạn tạo ra các hiệu ứng chuyển động mượt mà, thu hút sự chú ý và nâng cao trải nghiệm người dùng.

+Nguyên lí hoạt động

Animations hoạt động bằng cách xác định một chuỗi keyframes, nơi bạn mô tả trạng thái của một hoặc nhiều thuộc tính CSS tại các thời điểm cụ thể. Trình duyệt sau đó sẽ tự động chuyển đổi giữa các trạng thái này theo thời gian tạo thành hiệu ứng chuyển động.

+Cấu trúc

1. **Keyframes:**
   * Sử dụng @keyframes với tên animation mong muốn.
   * Xác định các khung hình chính (keyframes) với phần trăm hoặc từ khóa from và to.
   * Trong mỗi keyframe, khai báo các thuộc tính CSS và giá trị tương ứng.

## 3.12. SCSS

SCSS (Sassy CSS) là một ngôn ngữ tiền xử lý CSS phổ biến, giúp bạn viết CSS theo phong cách lập trình, mang lại nhiều lợi ích và cải thiện hiệu quả công việc.

Tính năng nổi bật:

* Biến và Mixins: Giúp bạn tái sử dụng code, giảm thiểu lặp lại và dễ dàng quản lý code CSS.
* Nesting: Viết code CSS theo cấu trúc lồng nhau, giúp code gọn gàng, trực quan và dễ hiểu hơn.
* Operations: Thực hiện các phép toán toán học và logic trong code CSS, giúp tạo hiệu ứng phức tạp dễ dàng hơn.
* Functions: Tạo và sử dụng các hàm tùy chỉnh để tối ưu hóa code CSS.
* Loops: Lặp lại các đoạn code CSS dựa trên điều kiện, giúp tiết kiệm thời gian và công sức.

Lợi ích:

* Tăng hiệu quả: Viết code CSS nhanh hơn, dễ dàng hơn và ít lỗi hơn.
* Bảo trì: Dễ dàng quản lý, sửa đổi và cập nhật code CSS.
* Tính linh hoạt: Tạo ra các giao diện web phức tạp và hiệu ứng đẹp mắt.
* Khả năng mở rộng: Hỗ trợ nhiều thư viện và plugin để tăng cường chức năng.

Hạn chế:

* Cần học thêm cú pháp mới: SCSS có cú pháp riêng biệt so với CSS.
* Tương thích trình duyệt: Cần biên dịch SCSS sang CSS trước khi sử dụng trên trình duyệt.

Công cụ:

* LibSass: Thư viện mã nguồn mở để biên dịch SCSS sang CSS.
* Sass compilers: Các trình biên dịch SCSS trực tuyến hoặc tích hợp trong IDE.

## 3.13. SASS

SASS là một ngôn ngữ tiền xử lý CSS phổ biến, giúp bạn viết CSS theo phong cách lập trình, mang lại nhiều lợi ích và cải thiện hiệu quả công việc.

Tên gọi:

* SCSS: Tên đầy đủ là Syntactically Awesome StyleSheet, là tên chính thức của ngôn ngữ.
* SASS: Viết tắt của SCSS, thường được sử dụng như một cách gọi ngắn gọn và phổ biến hơn.

Cú pháp:

* SCSS sử dụng cú pháp dựa trên CSS, nhưng có thêm các tính năng nâng cao như:
  + Biến và Mixins: Giúp bạn tái sử dụng code, giảm thiểu lặp lại và dễ dàng quản lý code CSS.
  + Nesting: Viết code CSS theo cấu trúc lồng nhau, giúp code gọn gàng, trực quan và dễ hiểu hơn.
  + Operations: Thực hiện các phép toán toán học và logic trong code CSS, giúp tạo hiệu ứng phức tạp dễ dàng hơn.
  + Functions: Tạo và sử dụng các hàm tùy chỉnh để tối ưu hóa code CSS.
  + Loops: Lặp lại các đoạn code CSS dựa trên điều kiện, giúp tiết kiệm thời gian và công sức.

Lợi ích:

* Tăng hiệu quả: Viết code CSS nhanh hơn, dễ dàng hơn và ít lỗi hơn.
* Bảo trì: Dễ dàng quản lý, sửa đổi và cập nhật code CSS.
* Tính linh hoạt: Tạo ra các giao diện web phức tạp và hiệu ứng đẹp mắt.
* Khả năng mở rộng: Hỗ trợ nhiều thư viện và plugin để tăng cường chức năng.

Hạn chế:

* Cần học thêm cú pháp mới: SCSS có cú pháp riêng biệt so với CSS.
* Tương thích trình duyệt: Cần biên dịch SCSS sang CSS trước khi sử dụng trên trình duyệt.

Công cụ:

* LibSass: Thư viện mã nguồn mở để biên dịch SCSS sang CSS.
* Sass compilers: Các trình biên dịch SCSS trực tuyến hoặc tích hợp trong IDE.

## \* Kết luận chương 3

CSS (Cascading Style Sheets):

* Ngôn ngữ định kiểu cho các trang web, giúp kiểm soát giao diện và bố cục.
* Xác định màu sắc, phông chữ, kích thước, vị trí, v.v. của các element HTML.
* Giúp tạo ra giao diện web đẹp mắt, thu hút và dễ sử dụng.

CSS3:

* Phiên bản nâng cao của CSS, mang đến nhiều tính năng mới mạnh mẽ.
* Hỗ trợ các hiệu ứng chuyển động, animations, bố cục đa cột, v.v.
* Giúp tạo ra giao diện web hiện đại, tương tác và hiệu quả.

Ưu điểm của CSS3:

* Nâng cao khả năng sáng tạo trong thiết kế web.
* Tạo ra giao diện web mượt mà, năng động và thu hút.
* Cải thiện trải nghiệm người dùng.
* Hỗ trợ nhiều tính năng mới và tiên tiến.

Hạn chế của CSS3:

* Khả năng tương thích trình duyệt có thể khác nhau.
* Cần có kiến thức chuyên môn để sử dụng hiệu quả.
* Sử dụng quá nhiều hiệu ứng có thể ảnh hưởng đến hiệu suất.

Kết luận:

CSS và CSS3 là những công cụ thiết yếu để tạo ra các trang web đẹp mắt, hiện đại và dễ sử dụng. Việc sử dụng CSS3 hiệu quả giúp bạn nâng cao khả năng sáng tạo và tạo ra giao diện web ấn tượng, thu hút người dùng.

Lời khuyên:

* Nắm vững kiến thức cơ bản về CSS.
* Tìm hiểu và áp dụng các tính năng CSS3 phù hợp với nhu cầu.
* Kiểm tra khả năng tương thích trình duyệt.
* Sử dụng CSS3 một cách hợp lý để đảm bảo hiệu suất tốt nhất.