Pathwise
http://localhost:3000/roadmaps/html

Quét để xem sơ đồ trực tuyến

FRONTEND

HTML thực chiến: Xây nền tảng Web chuẩn, dễ tiếp cận trong thời đại AI

Lộ trình HTML từ con số 0 đến dự án website đa trang: cú pháp và DOM, nội dung, liên kết, hình ảnh, biểu mẫu, semantic HTML, accessibility, SEO, hiệu năng, bảo mật cơ bản, kiểm thử và quy trình sử dụng AI có kiểm chứng. Học theo năng lực, không ép tiến độ theo ngày.

Cơ bản · 9 giai đoạn · 42 chủ đề · Cập nhật 04/11/2026

Tài liệu nguồn chính: https://roadmap.sh/html

MỤC TIÊU

- Hiểu HTML là gì, trình duyệt xây dựng DOM như thế nào và HTML khác CSS/JavaScript ra sao
- Tự tạo trang HTML đúng chuẩn, tổ chức nội dung, liên kết và tài nguyên trên website nhiều trang
- Làm bảng dữ liệu và biểu mẫu đúng ngữ nghĩa, hiểu luồng GET/POST và giới hạn xác thực phía client
- Áp dụng semantic HTML và accessibility: heading, landmarks, alt, labels, keyboard, ARIA có chọn lọc
- Biết tối ưu metadata, SEO cơ bản, ảnh, tốc độ tải và giảm rủi ro với HTML/iframe/form
- Sử dụng AI để học, đề xuất và review HTML nhưng tự kiểm chứng bằng MDN, validator và trình duyệt
- Hoàn thiện website sự kiện ít nhất 5 trang, có kiểm thử, báo cáo và khả năng triển khai

CÁC GIAI ĐOẠN

Giai đoạn 1: 01 · Nhập môn Web và HTML

Học cách web vận hành, thiết lập môi trường và phân biệt HTML với CSS, JavaScript trước khi gõ mã. (4 chủ đề)

Giai đoạn 2: 02 · Cú pháp và tài liệu HTML hoàn chỉnh

Từ cấu trúc chuẩn tới quy tắc lồng thẻ, ký tự đặc biệt và metadata. (4 chủ đề)

Giai đoạn 3: 03 · Nội dung, văn bản và cấu trúc đọc hiểu

Xây bài viết dễ đọc, dễ điều hướng và có ý nghĩa với máy đọc màn hình. (4 chủ đề)

Giai đoạn 4: 04 · Liên kết, URL và điều hướng

Xây website nhiều trang, điều hướng bằng liên kết hoạt động và an toàn. (4 chủ đề)

Giai đoạn 5: 05 · Ảnh, đa phương tiện và nội dung nhúng

Làm chủ ngữ nghĩa ảnh, media, hiệu năng và nhúng nội dung đáng tin cậy. (4 chủ đề)

Giai đoạn 6: 06 · Bảng dữ liệu và biểu mẫu HTML

Xây bảng có ý nghĩa và biểu mẫu gửi dữ liệu rõ ràng, dễ dùng và an toàn. (7 chủ đề)

Giai đoạn 7: 07 · Semantic HTML và khả năng tiếp cận

Thay div soup bằng cấu trúc có ý nghĩa và phục vụ người dùng thực tế. (6 chủ đề)

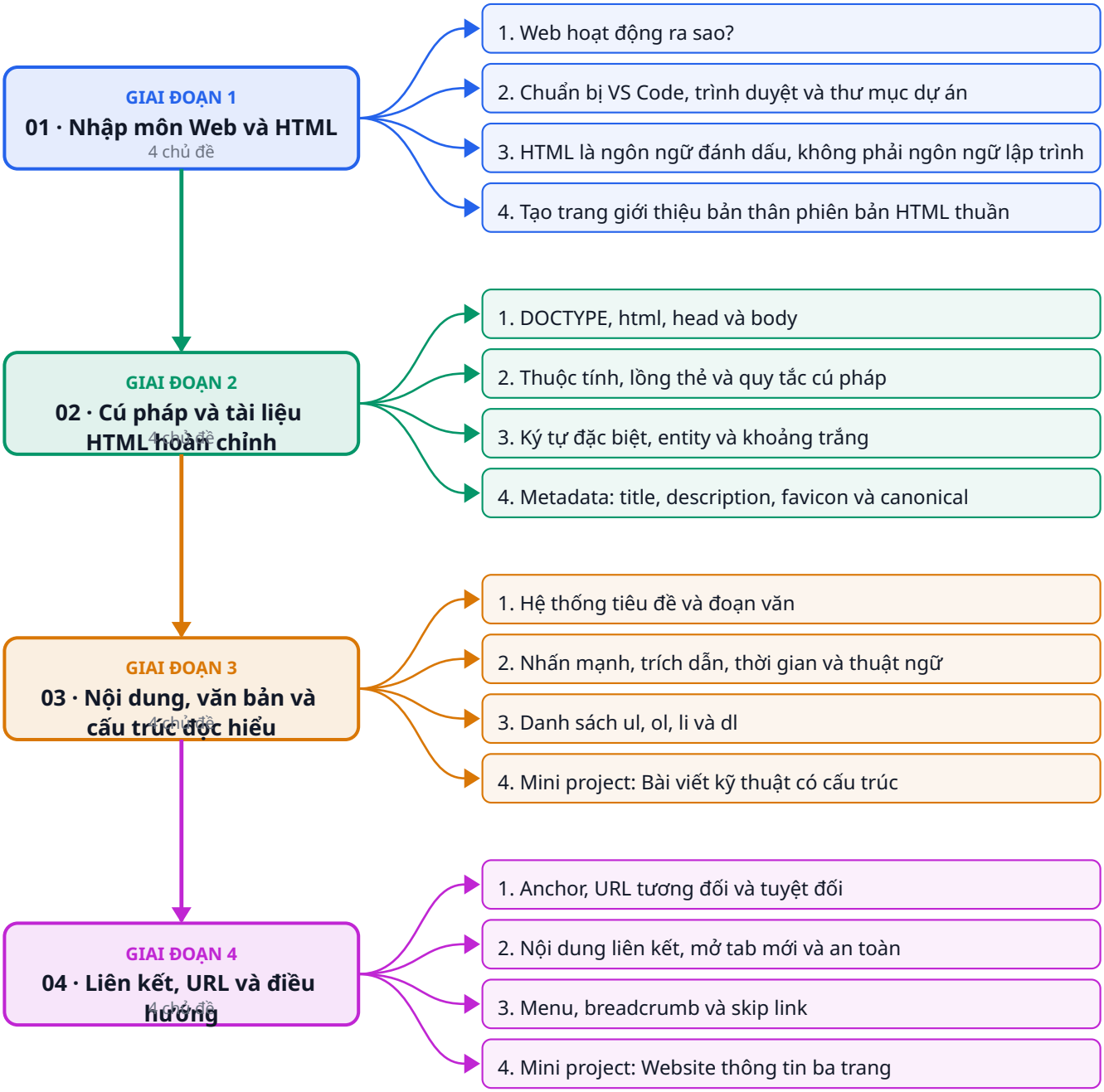
Giai đoạn 8: 08 · SEO, hiệu năng và an toàn nền tảng

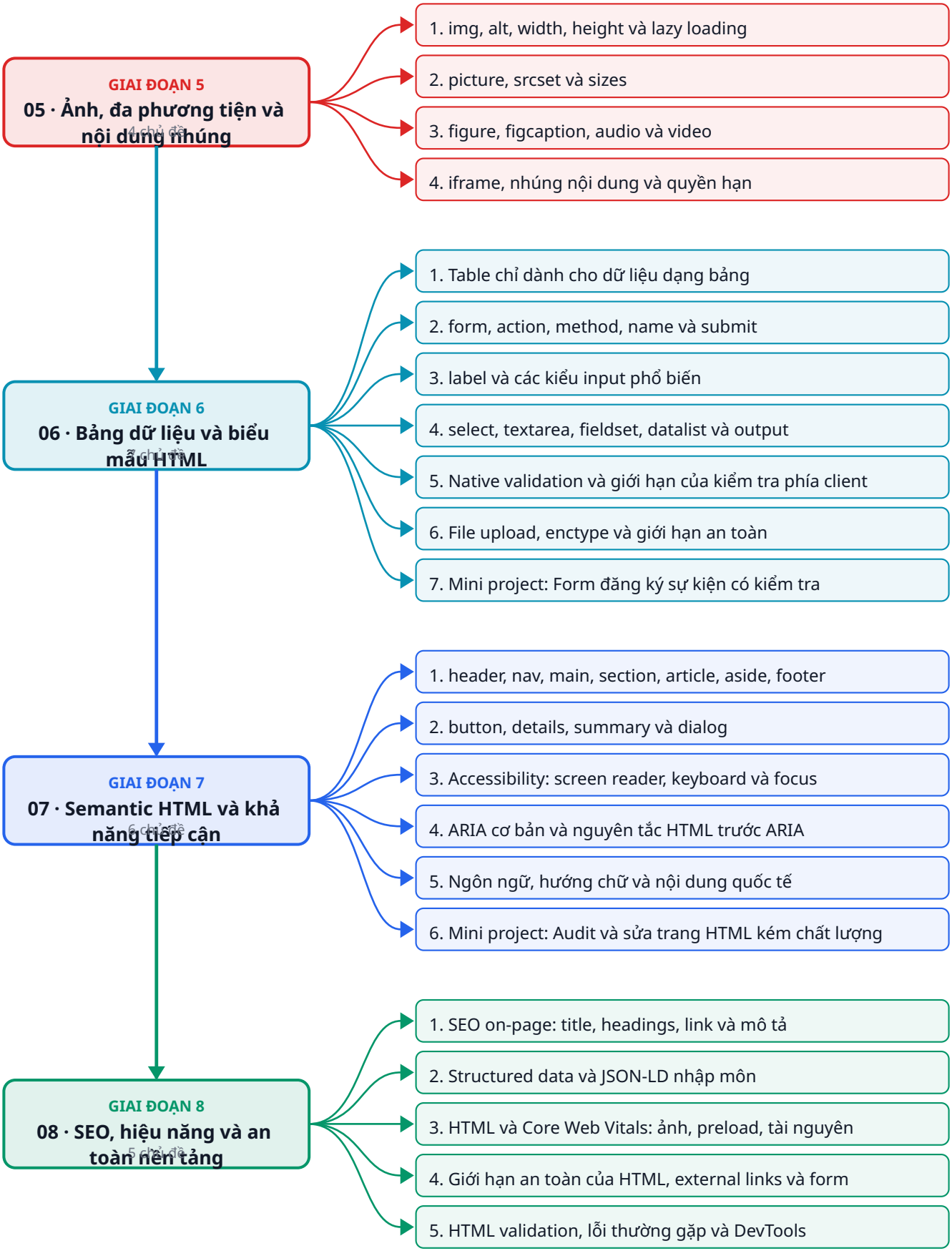
Kết nối HTML với tìm kiếm, tốc độ tải, chính sách bảo mật và dữ liệu người dùng. (5 chủ đề)

Giai đoạn 9: 09 · Quy trình thực tế, AI hỗ trợ và dự án cuối

Ứng dụng HTML trong dự án nhóm, framework và xây sản phẩm có thể bàn giao. (4 chủ đề)

SƠ ĐỒ LỘ TRÌNH




GIAI ĐOẠN 5
05 · Ảnh, đa phương tiện và nội dung nhúng
4 chủ đề
1. img, alt, width, height và lazy loading
2. picture, srcset và sizes
3. figure, figcaption, audio và video
4. iframe, nhúng nội dung và quyền hạn
GIAI ĐOẠN 6
06 · Bảng dữ liệu và biểu mẫu HTML
7 chủ đề
1. Table chỉ dành cho dữ liệu dạng bảng
2. form, action, method, name và submit
3. label và các kiểu input phổ biến
4. select, textarea, fieldset, datalist và output
5. Native validation và giới hạn của kiểm tra phía client
6. File upload, enctype và giới hạn an toàn
7. Mini project: Form đăng ký sự kiện có kiểm tra
GIAI ĐOẠN 7
07 · Semantic HTML và khả năng tiếp cận
6 chủ đề
1. header, nav, main, section, article, aside, footer
2. button, details, summary và dialog
3. Accessibility: screen reader, keyboard và focus
4. ARIA cơ bản và nguyên tắc HTML trước ARIA
5. Ngôn ngữ, hướng chữ và nội dung quốc tế
6. Mini project: Audit và sửa trang HTML kém chất lượng
GIAI ĐOẠN 8
08 · SEO, hiệu năng và an toàn nền tảng
5 chủ đề
1. SEO on-page: title, headings, link và mô tả
2. Structured data và JSON-LD nhập môn
3. HTML và Core Web Vitals: ảnh, preload, tài nguyên
4. Giới hạn an toàn của HTML, external links và form
5. HTML validation, lỗi thường gặp và DevTools


GIAI ĐOẠN 9
09 · Quy trình thực tế, AI hỗ trợ và dự án cuối
4 chủ đề
1. HTML trong template, React, Next.js và SSR
2. Dùng AI học HTML và review code mà không lệ thuộc
3. Capstone: Website thông tin sự kiện 5 trang HTML thuần
4. Học tiếp CSS, JavaScript, Git và Web Standards

GIAI ĐOẠN 1

01 · Nhập môn Web và HTML

Học cách web vận hành, thiết lập môi trường và phân biệt HTML với CSS, JavaScript trước khi gõ mã.

1. Web hoạt động ra sao?

Hiểu hành trình từ URL đến tài liệu hiển thị.

Mục tiêu học tập

- Hiểu và giải thích được URL trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "Web hoạt động ra sao?" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

URL · HTTP/HTTPS · Browser · Server · Response · DOM

Giải thích chi tiết

Vì sao cần học trong thực tế?

Hiểu hành trình từ URL đến tài liệu hiển thị. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- Trình duyệt gửi HTTP request đến máy chủ, nhận tài liệu HTML và tài nguyên liên quan.
- HTML mô tả cấu trúc; CSS trình bày; JavaScript điều khiển hành vi; ba vai trò khác nhau.
- View Source là HTML máy chủ trả về; Elements trong DevTools là DOM hiện tại, có thể bị JavaScript thay đổi.
- Không phải URL nào cũng kết thúc bằng .html; máy chủ có thể sinh HTML động.

Ví dụ HTML

```
<!doctype html>
<html lang="vi">
<head><meta charset="utf-8"><title>Xin chào Web</title></head>
<body><h1>Trang đầu tiên</h1></body>
</html>
```

Thực hành từng bước

1. Mở một website và kiểm tra tab Network để tìm document request.
2. So sánh View Page Source với thẻ Elements.
3. Ghi lại ba ví dụ về HTML/CSS/JS trên trang.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài "Web hoạt động ra sao?". Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Giải thích được một trang được tải bằng 5 bước không dùng thuật ngữ mơ hồ. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<!doctype html>
<html lang="vi">
<head><meta charset="utf-8"><title>Xin chào Web</title></head>
<body><h1>Trang đầu tiên</h1></body>
</html>
```

Hiểu hành trình từ URL đến tài liệu hiển thị.

Checklist

- [ ] Đã đọc và giải thích được các khái niệm: URL, HTTP/HTTPS, Browser, Server, Response, DOM
- [ ] Đã làm: Mở một website và kiểm tra tab Network để tìm document request.
- [ ] Đã làm: So sánh View Page Source với thẻ Elements.
- [ ] Đã làm: Ghi lại ba ví dụ về HTML/CSS/JS trên trang.
- [ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
- [ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
- [ ] Hoàn thành: Giải thích được một trang được tải bằng 5 bước không dùng thuật ngữ mơ hồ.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Learn_web_development/Getting_started/Web_standards/How_the_web_works

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

2. Chuẩn bị VS Code, trình duyệt và thư mục dự án

Tạo môi trường học bằng file HTML thuần, chưa dùng framework.

Mục tiêu học tập

- Hiểu và giải thích được VS Code trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "Chuẩn bị VS Code, trình duyệt và thư mục dự án" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

VS Code · File Extension · Browser DevTools · Local File · Live Server

Giải thích chi tiết

Vì sao cần học trong thực tế?

Tạo môi trường học bằng file HTML thuần, chưa dùng framework. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- File HTML có phần mở rộng .html, không phải .html.txt.
- Dùng trình duyệt hiện đại và DevTools để đọc cây DOM, thuộc tính và lỗi.
- Có thể mở file trực tiếp hoặc dùng local development server; tránh nhầm file:// với máy chủ HTTP.
- Đặt tên file, ảnh, thư mục nhất quán; tránh dấu cách và ký tự khó xử lý trong URL.

Ví dụ HTML

```
<!-- html-lab/
  index.html
  assets/images/
  pages/about.html
-->
```

Thực hành từng bước

1. Tạo thư mục html-lab với index.html, assets/images.
2. Mở file trong trình duyệt và sửa tiêu đề.
3. Inspect h1 và thử sửa chữ trong DevTools, sau đó reload để nhận biết thay đổi tạm thời.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “Chuẩn bị VS Code, trình duyệt và thư mục dự án”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Tự tạo và mở trang HTML mà không cần AI tạo dự án hộ. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<!-- html-lab/
  index.html
  assets/images/
  pages/about.html
-->
```

Tạo môi trường học bằng file HTML thuần, chưa dùng framework.

Checklist

- [ ] Đã đọc và giải thích được các khái niệm: VS Code, File Extension, Browser DevTools, Local File, Live Server
- [ ] Đã làm: Tạo thư mục html-lab với index.html, assets/images.
- [ ] Đã làm: Mở file trong trình duyệt và sửa tiêu đề.
- [ ] Đã làm: Inspect h1 và thử sửa chữ trong DevTools, sau đó reload để nhận biết thay đổi tạm thời.
- [ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
- [ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
- [ ] Hoàn thành: Tự tạo và mở trang HTML mà không cần AI tạo dự án hộ.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Learn_web_development/Getting_started/Environment_setup/Installing_software

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

Web hoạt động ra sao?

3. HTML là ngôn ngữ đánh dấu, không phải ngôn ngữ lập trình

Nắm phần tử, thẻ, nội dung, thuộc tính và DOM.

Mục tiêu học tập

- Hiểu và giải thích được Element trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "HTML là ngôn ngữ đánh dấu, không phải ngôn ngữ lập trình" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

Element · Tag · Attribute · DOM Tree · Nesting · Void Element

Giải thích chi tiết

Vì sao cần học trong thực tế?

Nắm phần tử, thẻ, nội dung, thuộc tính và DOM. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- Phần tử gồm thẻ mở, nội dung, thẻ đóng trong trường hợp thông thường.
- Thuộc tính mô tả thêm thông tin cho phần tử; đặt trong thẻ mở.
- DOM là cây nút do trình duyệt xây dựng từ HTML; mã nguồn sai có thể được trình duyệt tự sửa khi parse.
- Phần tử void như img, br, input không có thẻ đóng; không viết nội dung con cho chúng.
- HTML không tự chứa vòng lặp hoặc điều kiện thực thi như JavaScript.

Ví dụ HTML

```
<article class="post">
  <h2>Học HTML</h2>
  <p>Bắt đầu từ <strong>ngữ nghĩa</strong>.</p>
  <a href="/about.html">Giới thiệu</a>
</article>
```

Thực hành từng bước

1. Viết một h1, hai p và một a, rồi chỉ ra từng thẻ và thuộc tính.
2. Dùng DevTools để quan sát quan hệ cha-con.
3. Viết sai cấu trúc có chủ ý và xem DOM đã được sửa thế nào.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “HTML là ngôn ngữ đánh dấu, không phải ngôn ngữ lập trình”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Giải thích chính xác tag, element, attribute, DOM và nesting bằng ví dụ. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<article class="post">
  <h2>Học HTML</h2>
  <p>Bắt đầu từ <strong>ngữ nghĩa</strong>.</p>
  <a href="/about.html">Giới thiệu</a>
</article>
```

Nắm phần tử, thẻ, nội dung, thuộc tính và DOM.

Checklist

[ ] Đã đọc và giải thích được các khái niệm: Element, Tag, Attribute, DOM Tree, Nesting, Void Element

[ ] Đã làm: Viết một h1, hai p và một a, rồi chỉ ra từng thẻ và thuộc tính.

[ ] Đã làm: Dùng DevTools để quan sát quan hệ cha-con.

[ ] Đã làm: Viết sai cấu trúc có chủ ý và xem DOM đã được sửa thế nào.

[ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp

[ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc

[ ] Hoàn thành: Giải thích chính xác tag, element, attribute, DOM và nesting bằng ví dụ.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Structuring_content

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

Chuẩn bị VS Code, trình duyệt và thư mục dự án

4. Tạo trang giới thiệu bản thân phiên bản HTML thuần

Dự án nhỏ đầu tiên, chỉ dùng các phần tử đã hiểu.

Mục tiêu học tập

- Hiểu và giải thích được HTML Document trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "Tạo trang giới thiệu bản thân phiên bản HTML thuần" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

HTML Document · Heading · Paragraph · Link · Image

Giải thích chi tiết

Vì sao cần học trong thực tế?

Dự án nhỏ đầu tiên, chỉ dùng các phần tử đã hiểu. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- Trang HTML là tài liệu có cấu trúc, không phải hình ảnh được ghép từ thẻ.
- Nội dung cần có tiêu đề và các đoạn đọc được khi tắt CSS.
- Dùng liên kết cho điều hướng, không sử dụng nút giả chỉ để đi trang khác.

Ví dụ HTML

```
<h1>Xin chào, mình đang học HTML</h1>
<p>Mục tiêu: tự xây website có cấu trúc tốt.</p>
<a href="https://developer.mozilla.org/">Tài liệu MDN</a>
```

Thực hành từng bước

1. Làm trang giới thiệu gồm tên, mục tiêu học, ba sở thích, liên kết học tập.
2. Kiểm tra bằng View Source và DevTools.
3. Nhờ một người khác đọc trang không có CSS để xem họ có hiểu bố cục.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài "Tạo trang giới thiệu bản thân phiên bản HTML thuần". Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Nộp một file index.html mở được, không lỗi tài nguyên. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<h1>Xin chào, mình đang học HTML</h1>
<p>Mục tiêu: tự xây website có cấu trúc tốt.</p>
<a href="https://developer.mozilla.org/">Tài liệu MDN</a>
```

Dự án nhỏ đầu tiên, chỉ dùng các phần tử đã hiểu.

Checklist

[ ] Đã đọc và giải thích được các khái niệm: HTML Document, Heading, Paragraph, Link, Image

[ ] Đã làm: Làm trang giới thiệu gồm tên, mục tiêu học, ba sở thích, liên kết học tập.

[ ] Đã làm: Kiểm tra bằng View Source và DevTools.

[ ] Đã làm: Nhờ một người khác đọc trang không có CSS để xem họ có hiểu bố cục.

[ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp

[ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc

[ ] Hoàn thành: Nộp một file index.html mở được, không lỗi tài nguyên.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Structuring_content

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

HTML là ngôn ngữ đánh dấu, không phải ngôn ngữ lập trình

GIAI ĐOẠN 2

02 · Cú pháp và tài liệu HTML hoàn chỉnh

Từ cấu trúc chuẩn tới quy tắc lồng thẻ, ký tự đặc biệt và metadata.

1. DOCTYPE, html, head và body

Viết bộ khung HTML hiện đại hợp lệ.

Mục tiêu học tập

- Hiểu và giải thích được DOCTYPE trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "DOCTYPE, html, head và body" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

DOCTYPE · html lang · head · body · charset · viewport · title

Giải thích chi tiết

Vì sao cần học trong thực tế?

Viết bộ khung HTML hiện đại hợp lệ. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- <!doctype html> yêu cầu chế độ tiêu chuẩn; nó không phải phần tử HTML.
- html lang giúp công cụ hỗ trợ xác định ngôn ngữ chính.
- head chứa metadata, liên kết tài nguyên; body chứa nội dung trang hiển thị.
- meta charset="utf-8" giúp hiển thị tiếng Việt chính xác; viewport hỗ trợ thiết bị di động.
- title là tên hiển thị trên tab và ngữ cảnh kết quả tìm kiếm.

Ví dụ HTML

```
<!doctype html>
<html lang="vi">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>Học HTML | Trang chủ</title>
</head>
<body>
  <h1>Học HTML đúng cách</h1>
</body>
</html>
```

Thực hành từng bước

1. Tự gõ skeleton không dùng snippet.
2. Đổi lang thành vi, thêm title và viewport.
3. Thử bỏ charset rồi quan sát/giải thích rủi ro mã hóa.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài "DOCTYPE, html, head và body". Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Tự tái tạo skeleton chuẩn từ trang trắng. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<!doctype html>
<html lang="vi">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>Học HTML | Trang chủ</title>
</head>
<body>
  <h1>Học HTML đúng cách</h1>
</body>
</html>
```

Viết bộ khung HTML hiện đại hợp lệ.

Checklist

[ ] Đã đọc và giải thích được các khái niệm: DOCTYPE, html lang, head, body, charset, viewport, title

[ ] Đã làm: Tự gõ skeleton không dùng snippet.

[ ] Đã làm: Đổi lang thành vi, thêm title và viewport.

[ ] Đã làm: Thử bỏ charset rồi quan sát/giải thích rủi ro mã hóa.

[ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp

[ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc

[ ] Hoàn thành: Tự tái tạo skeleton chuẩn từ trang trắng.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Structuring_content/Webpage_metadata

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

Tạo trang giới thiệu bản thân phiên bản HTML thuần

---

2. Thuộc tính, lồng thẻ và quy tắc cú pháp

Tránh lỗi DOM âm thầm tự sửa.

Mục tiêu học tập

- Hiểu và giải thích được Boolean Attribute trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "Thuộc tính, lồng thẻ và quy tắc cú pháp" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

Boolean Attribute · Global Attribute · Nesting · Void Element · Comments

Giải thích chi tiết

Vì sao cần học trong thực tế?

Tránh lỗi DOM âm thầm tự sửa. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- Một số thuộc tính có giá trị như id, class, href, lang; boolean attribute như required, disabled hoạt động theo sự hiện diện.
- Giá trị thuộc tính nên đặt trong dấu nháy, đặc biệt khi có dấu cách.
- Lồng thẻ phải đúng quan hệ phần tử; không tùy tiện đặt block trong p.
- id phải duy nhất trong cùng tài liệu; class có thể lặp lại.
- Comment không hiển thị cho người dùng nhưng vẫn có thể xem trong mã nguồn, không chứa secrets.

Ví dụ HTML

```
<p class="notice" id="intro">
  Nội dung <strong>quan trọng</strong> ở đây.
</p>
<input type="email" name="email" required>
```

Thực hành từng bước

1. Tạo ví dụ đúng và sai về lồng strong/em.
2. Thử required và required="false" để thấy boolean attribute vẫn có hiệu lực.
3. Chạy HTML validator và sửa lỗi.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “Thuộc tính, lồng thẻ và quy tắc cú pháp”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.
- required="false" vẫn coi là required; để tắt phải bỏ thuộc tính.

Tiêu chí hoàn thành

Tìm và sửa được lỗi lồng thẻ bằng validator. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<p class="notice" id="intro">
  Nội dung <strong>quan trọng</strong> ở đây.
</p>
<input type="email" name="email" required>
```

Tránh lỗi DOM âm thầm tự sửa.

Checklist

- [ ] Đã đọc và giải thích được các khái niệm: Boolean Attribute, Global Attribute, Nesting, Void Element, Comments
- [ ] Đã làm: Tạo ví dụ đúng và sai về lồng strong/em.
- [ ] Đã làm: Thử required và required="false" để thấy boolean attribute vẫn có hiệu lực.
- [ ] Đã làm: Chạy HTML validator và sửa lỗi.
- [ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
- [ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
- [ ] Hoàn thành: Tìm và sửa được lỗi lồng thẻ bằng validator.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Structuring_content/Basic_HTML_syntax

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

DOCTYPE, html, head và body

3. Ký tự đặc biệt, entity và khoảng trắng

Hiểu vì sao ký tự đôi khi cần escape và cách trình duyệt gom whitespace.

Mục tiêu học tập

- Hiểu và giải thích được Character Reference trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "Ký tự đặc biệt, entity và khoảng trắng" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

Character Reference · Entity · Whitespace · Nonbreaking Space · Preformatted Text

Giải thích chi tiết

Vì sao cần học trong thực tế?

Hiểu vì sao ký tự đôi khi cần escape và cách trình duyệt gom whitespace. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- Trong văn bản HTML, dùng < để hiển thị dấu < khi cần tránh parse thành thẻ.
- & hiển thị &, " hiển thị dấu nháy trong một số ngữ cảnh.
- Nhiều dấu cách và xuống dòng thường được gom lại trong luồng văn bản thường.
- pre giữ whitespace theo nguồn; code chỉ đánh dấu nội dung mã, không tự lưu khoảng trắng.
- Không dùng hàng loạt   để căn chỉnh giao diện; đó là việc của CSS.

Ví dụ HTML

```
<p>Viết thẻ &lt;strong&gt; trong tài liệu.</p>
<pre><code>const x = 1;
const y = 2;</code></pre>
```

Thực hành từng bước

1. Viết đoạn chứa dưới dạng văn bản.
2. So sánh p, pre và code.
3. Chỉnh khoảng trắng bằng CSS ở bài sau thay vì spam entity.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài "Ký tự đặc biệt, entity và khoảng trắng". Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Hiển thị ký tự đặc biệt và đoạn code đúng mà không phá HTML. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<p>Viết thẻ &lt;strong&gt; trong tài liệu.</p>
<pre><code>const x = 1;
const y = 2;</code></pre>
```

Hiểu vì sao ký tự đôi khi cần escape và cách trình duyệt gom whitespace.

Checklist

- [ ] Đã đọc và giải thích được các khái niệm: Character Reference, Entity, Whitespace, Nonbreaking Space, Preformatted Text
- [ ] Đã làm: Viết đoạn chứa dưới dạng văn bản.
- [ ] Đã làm: So sánh p, pre và code.
- [ ] Đã làm: Chỉnh khoảng trắng bằng CSS ở bài sau thay vì spam entity.
- [ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
- [ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
- [ ] Hoàn thành: Hiển thị ký tự đặc biệt và đoạn code đúng mà không phá HTML.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Structuring_content

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

Thuộc tính, lồng thẻ và quy tắc cú pháp

---

4. Metadata: title, description, favicon và canonical

Tách dữ liệu mô tả khỏi nội dung chính.

Mục tiêu học tập

- Hiểu và giải thích được meta description trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề “Metadata: title, description, favicon và canonical” mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

meta description · favicon · canonical · viewport · robots · Open Graph

Giải thích chi tiết

Vì sao cần học trong thực tế?

Tách dữ liệu mô tả khỏi nội dung chính. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- title và description nên mô tả cụ thể mục đích từng trang.
- Favicon là biểu tượng của website; link rel="icon" khai báo tài nguyên.
- canonical là gợi ý URL ưu tiên khi có nội dung trùng lặp, không phải lệnh ép index.
- Open Graph giúp ứng dụng hỗ trợ tạo bản xem trước khi chia sẻ.
- Metadata có thể do framework tạo nhưng vẫn phải hiểu HTML kết quả.

Ví dụ HTML

```
<title>Giới thiệu | HTML Lab</title>
<meta name="description" content="Tìm hiểu dự án HTML Lab và các bài học thực hành.">
<link rel="icon" href="/favicon.ico">
<link rel="canonical" href="https://example.com/about">
```

Thực hành từng bước

1. Tạo title và description riêng cho trang chủ, giới thiệu.
2. Thêm favicon hợp lệ và xem tab trình duyệt.
3. Kiểm tra thẻ meta bằng View Source.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “Metadata: title, description, favicon và canonical”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Hai trang có title, description riêng và metadata không mâu thuẫn. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<title>Giới thiệu | HTML Lab</title>
<meta name="description" content="Tìm hiểu dự án HTML Lab và các bài học thực hành.">
<link rel="icon" href="/favicon.ico">
<link rel="canonical" href="https://example.com/about">
```

Tách dữ liệu mô tả khỏi nội dung chính.

Checklist

- [ ] Đã đọc và giải thích được các khái niệm: meta description, favicon, canonical, viewport, robots, Open Graph
- [ ] Đã làm: Tạo title và description riêng cho trang chủ, giới thiệu.
- [ ] Đã làm: Thêm favicon hợp lệ và xem tab trình duyệt.
- [ ] Đã làm: Kiểm tra thẻ meta bằng View Source.
- [ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
- [ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
- [ ] Hoàn thành: Hai trang có title, description riêng và metadata không mâu thuẫn.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Structuring_content/Webpage_metadata

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

Ký tự đặc biệt, entity và khoảng trắng

GIAI ĐOẠN 3

03 · Nội dung, văn bản và cấu trúc đọc hiểu

Xây bài viết dễ đọc, dễ điều hướng và có ý nghĩa với máy đọc màn hình.

1. Hệ thống tiêu đề và đoạn văn

Tổ chức thứ bậc nội dung bằng h1–h6, p.

Mục tiêu học tập

- Hiểu và giải thích được h1-h6 trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "Hệ thống tiêu đề và đoạn văn" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

h1-h6 · Heading Hierarchy · p · Document Outline

Giải thích chi tiết

Vì sao cần học trong thực tế?

Tổ chức thứ bậc nội dung bằng h1–h6, p. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- Heading tạo thứ bậc nội dung, không chọn h3 vì nó trông nhỏ hơn h2.
- Thông lệ nên có tiêu đề chính rõ ràng và hệ thống cấp bậc không nhảy vô lý.
- p dành cho đoạn văn; br chỉ xuống dòng khi nội dung thực sự cần ngắt dòng.
- Một heading cần mô tả nội dung theo sau, không dùng làm văn bản trang trí.

Ví dụ HTML

```
<main>
  <h1>Học HTML</h1>
  <section>
    <h2>Cấu trúc trang</h2>
    <h3>Thẻ tiêu đề</h3>
    <p>Nội dung giải thích.</p>
  </section>
</main>
```

Thực hành từng bước

1. Viết bài hướng dẫn gồm h1, h2, h3 hợp lý.
2. Dùng công cụ đọc cấu trúc heading để rà soát.
3. Thử đổi h2 thành h4 và giải thích vì sao sai về cấu trúc.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “Hệ thống tiêu đề và đoạn văn”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Bài viết đọc rõ cấu trúc chỉ bằng danh sách heading. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<main>
  <h1>Học HTML</h1>
  <section>
    <h2>Cấu trúc trang</h2>
    <h3>Thẻ tiêu đề</h3>
    <p>Nội dung giải thích.</p>
  </section>
</main>
```

Tổ chức thứ bậc nội dung bằng h1–h6, p.

Checklist

[ ] Đã đọc và giải thích được các khái niệm: h1-h6, Heading Hierarchy, p, Document Outline
[ ] Đã làm: Viết bài hướng dẫn gồm h1, h2, h3 hợp lý.
[ ] Đã làm: Dùng công cụ đọc cấu trúc heading để rà soát.
[ ] Đã làm: Thử đổi h2 thành h4 và giải thích vì sao sai về cấu trúc.
[ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
[ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
[ ] Hoàn thành: Bài viết đọc rõ cấu trúc chỉ bằng danh sách heading.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML/Element/Heading_Elements

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

Metadata: title, description, favicon và canonical

---

2. Nhấn mạnh, trích dẫn, thời gian và thuật ngữ

Chọn strong/em, mark, code, abbr, time đúng mục đích.

Mục tiêu học tập

- Hiểu và giải thích được strong trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "Nhấn mạnh, trích dẫn, thời gian và thuật ngữ" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

strong · em · mark · small · abbr · time · code · blockquote · q

Giải thích chi tiết

Vì sao cần học trong thực tế?

Chọn strong/em, mark, code, abbr, time đúng mục đích. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- strong biểu thị tầm quan trọng; em biểu thị nhấn giọng; b/i có ý nghĩa trình bày/loại văn bản khác.
- blockquote dành cho trích dẫn dạng khối; q cho trích dẫn nội tuyến.
- time datetime gắn ngày giờ có thể đọc bằng máy.
- abbr title giải thích từ viết tắt; code đánh dấu mã nguồn; mark đánh dấu phần liên quan trong ngữ cảnh.
- Không dùng thẻ chỉ vì định dạng mặc định phù hợp mắt nhìn.

Ví dụ HTML

```
<p><abbr title="HyperText Markup Language">HTML</abbr> dùng để tạo <strong>cấu trúc</strong>
trang.</p>
<p>Cập nhật: <time datetime="2026-11-04">04/11/2026</time>.</p>
<blockquote cite="https://example.com/source"><p>Ví dụ trích dẫn.</p></blockquote>
```

Thực hành từng bước

1. Đánh dấu một bài giới thiệu thuật ngữ HTML bằng abbr và code.
2. Viết lời trích ngắn dùng q và trích dài dùng blockquote.
3. Thêm thời gian ISO vào time datetime.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài "Nhấn mạnh, trích dẫn, thời gian và thuật ngữ". Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Biện minh được tại sao dùng mỗi thẻ thay vì span. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<p><abbr title="HyperText Markup Language">HTML</abbr> dùng để tạo <strong>cấu trúc</strong>
trang.</p>
<p>Cập nhật: <time datetime="2026-11-04">04/11/2026</time>.</p>
<blockquote cite="https://example.com/source"><p>Ví dụ trích dẫn.</p></blockquote>
```

Chọn strong/em, mark, code, abbr, time đúng mục đích.

Checklist

[ ] Đã đọc và giải thích được các khái niệm: strong, em, mark, small, abbr, time, code, blockquote, q

[ ] Đã làm: Đánh dấu một bài giới thiệu thuật ngữ HTML bằng abbr và code.

[ ] Đã làm: Viết lời trích ngắn dùng q và trích dài dùng blockquote.

[ ] Đã làm: Thêm thời gian ISO vào time datetime.

[ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp

[ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc

[ ] Hoàn thành: Biện minh được tại sao dùng mỗi thẻ thay vì span.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Structuring_content

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

Hệ thống tiêu đề và đoạn văn

3. Danh sách ul, ol, li và dl

Biểu đạt tập hợp, thứ tự và cặp thuật ngữ–định nghĩa.

Mục tiêu học tập

- Hiểu và giải thích được ul trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "Danh sách ul, ol, li và dl" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

ul · ol · li · dl · dt · dd · Nested List

Giải thích chi tiết

Vì sao cần học trong thực tế?

Biểu đạt tập hợp, thứ tự và cặp thuật ngữ–định nghĩa. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- ul cho nhóm mục không phụ thuộc thứ tự; ol cho quy trình/thứ tự có nghĩa.
- li là mục con của ul/ol; danh sách lồng nhau nằm trong li.
- dl/dt/dd phù hợp danh sách mô tả, bảng thuật ngữ, metadata cặp tên–giá trị tùy ngữ cảnh.
- Không dùng danh sách chỉ vì muốn có dấu chấm trang trí.

Ví dụ HTML

```
<ol>
  <li>Tạo index.html</li>
  <li>Viết bộ khung</li>
  <li>Mở trình duyệt</li>
</ol>
<dl><dt>DOM</dt><dd>Mô hình cây của tài liệu.</dd></dl>
```

Thực hành từng bước

1. Viết checklist đồ dùng bằng ul.
2. Viết 5 bước tạo tài liệu bằng ol.
3. Tạo glossary 6 thuật ngữ bằng dl.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “Danh sách ul, ol, li và dl”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Phân biệt đúng ul, ol, dl trong ba tình huống. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<ol>
  <li>Tạo index.html</li>
  <li>Viết bộ khung</li>
  <li>Mở trình duyệt</li>
</ol>
<dl><dt>DOM</dt><dd>Mô hình cây của tài liệu.</dd></dl>
```

Biểu đạt tập hợp, thứ tự và cặp thuật ngữ–định nghĩa.

Checklist

[ ] Đã đọc và giải thích được các khái niệm: ul, ol, li, dl, dt, dd, Nested List

[ ] Đã làm: Viết checklist đồ dùng bằng ul.

[ ] Đã làm: Viết 5 bước tạo tài liệu bằng ol.

[ ] Đã làm: Tạo glossary 6 thuật ngữ bằng dl.

[ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp

[ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc

[ ] Hoàn thành: Phân biệt đúng ul, ol, dl trong ba tình huống.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Structuring_content

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

Nhấn mạnh, trích dẫn, thời gian và thuật ngữ

4. Mini project: Bài viết kỹ thuật có cấu trúc

Kết hợp tiêu đề, đoạn, danh sách, trích dẫn và metadata.

Mục tiêu học tập

- Hiểu và giải thích được Article trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "Mini project: Bài viết kỹ thuật có cấu trúc" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

Article · Reading Flow · Metadata · Content Hierarchy

Giải thích chi tiết

Vì sao cần học trong thực tế?

Kết hợp tiêu đề, đoạn, danh sách, trích dẫn và metadata. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- Bố cục bài viết phải hiểu được khi không có CSS.
- Dùng một article cho nội dung có thể tồn tại độc lập và heading theo cấp.
- Phần tóm tắt, danh sách bước, đoạn code và tham khảo phải dễ tìm.
- Không copy bài của người khác rồi chỉ đổi tiêu đề; nên dùng nội dung tự viết.

Ví dụ HTML

```
<article>
  <header><h1>Vì sao học HTML?</h1></header>
  <p>HTML tạo ngữ nghĩa cho nội dung.</p>
  <h2>Ba lợi ích</h2>
  <ul><li>Dễ tiếp cận</li><li>Đọc hiểu</li><li>Dễ bảo trì</li></ul>
</article>
```

Thực hành từng bước

1. Viết bài "HTML giúp gì cho website?" khoảng 500–800 từ tự viết.
2. Thêm mục lục bằng anchor sau khi học liên kết.
3. Đọc bằng chế độ reader nếu trình duyệt hỗ trợ.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “Mini project: Bài viết kỹ thuật có cấu trúc”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Bài viết dùng □10 phần tử HTML hợp ngữ cảnh, không có thẻ trang trí vô nghĩa. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<article>
  <header><h1>Vì sao học HTML?</h1></header>
  <p>HTML tạo ngữ nghĩa cho nội dung.</p>
  <h2>Ba lợi ích</h2>
  <ul><li>Dễ tiếp cận</li><li>Đọc hiểu</li><li>Dễ bảo trì</li></ul>
</article>
```

Kết hợp tiêu đề, đoạn, danh sách, trích dẫn và metadata.

Checklist

[ ] Đã đọc và giải thích được các khái niệm: Article, Reading Flow, Metadata, Content Hierarchy

[ ] Đã làm: Viết bài “HTML giúp gì cho website?” khoảng 500–800 từ tự viết.

[ ] Đã làm: Thêm mục lục bằng anchor sau khi học liên kết.

[ ] Đã làm: Đọc bằng chế độ reader nếu trình duyệt hỗ trợ.

[ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp

[ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc

[ ] Hoàn thành: Bài viết dùng □10 phần tử HTML hợp ngữ cảnh, không có thẻ trang trí vô nghĩa.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Structuring_content

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

Danh sách ul, ol, li và dl

GIAI ĐOẠN 4

04 · Liên kết, URL và điều hướng

Xây website nhiều trang, điều hướng bằng liên kết hoạt động và an toàn.

1. Anchor, URL tương đối và tuyệt đối

Dùng a href phù hợp với từng đích.

Mục tiêu học tập

- Hiểu và giải thích được a trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "Anchor, URL tương đối và tuyệt đối" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

a · href · Absolute URL · Relative URL · Root-relative URL · Fragment

Giải thích chi tiết

Vì sao cần học trong thực tế?

Dùng a href phù hợp với từng đích. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- a có href là liên kết; a không href không có cùng hành vi điều hướng.
- URL tuyệt đối có scheme/domain; URL tương đối giải quyết theo vị trí tài liệu hoặc base.
- /about.html là đường dẫn từ gốc origin; ./about.html từ thư mục hiện tại.
- #id trỏ tới phần tử cùng trang có id phù hợp.
- Không đưa đường dẫn file hệ điều hành cục bộ vào website xuất bản.

Ví dụ HTML

```
<nav aria-label="Điều hướng chính">
  <a href="/">Trang chủ</a>
  <a href="/about.html">Giới thiệu</a>
  <a href="/contact.html#form">Liên hệ</a>
</nav>
```

Thực hành từng bước

1. Tạo index.html, about.html và pages/contact.html.
2. Tạo link tương đối hai chiều rồi di chuyển thử file để xem hỏng ở đâu.
3. Thêm liên kết đến section bằng #id.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài "Anchor, URL tương đối và tuyệt đối". Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Không có link 404 trong bộ ba trang. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<nav aria-label="Điều hướng chính">
  <a href="/">Trang chủ</a>
  <a href="/about.html">Giới thiệu</a>
  <a href="/contact.html#form">Liên hệ</a>
</nav>
```

Dùng a href phù hợp với từng đích.

Checklist

- [ ] Đã đọc và giải thích được các khái niệm: a, href, Absolute URL, Relative URL, Root-relative URL, Fragment
- [ ] Đã làm: Tạo index.html, about.html và pages/contact.html.
- [ ] Đã làm: Tạo link tương đối hai chiều rồi di chuyển thử file để xem hỏng ở đâu.
- [ ] Đã làm: Thêm liên kết đến section bằng #id.
- [ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
- [ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
- [ ] Hoàn thành: Không có link 404 trong bộ ba trang.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Structuring_content/Creating_links

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

Mini project: Bài viết kỹ thuật có cấu trúc

2. Nội dung liên kết, mở tab mới và an toàn

Phân biệt link và button, tránh link gây hiểu lầm.

Mục tiêu học tập

- Hiểu và giải thích được Descriptive Link Text trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề “Nội dung liên kết, mở tab mới và an toàn” mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

Descriptive Link Text · target · rel · mailto · tel · download

Giải thích chi tiết

Vì sao cần học trong thực tế?

Phân biệt link và button, tránh link gây hiểu lầm. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- Link text phải hiểu được độc lập, tránh “bấm vào đây” lặp lại.
- target="_blank" mở ngữ cảnh mới; nên cân nhắc trải nghiệm và thông báo cho người dùng.
- rel="noopener noreferrer" có thể dùng khi liên kết ngoài cần hạn chế tham chiếu; trình duyệt hiện đại thường ngầm áp dụng noopener với _blank.
- mailto và tel là URI scheme chứ không phải địa chỉ web thông thường.
- a là điều hướng, button là kích hoạt hành động.

Ví dụ HTML

```
<a href="https://developer.mozilla.org/" target="_blank" rel="noopener noreferrer">Đọc tài
liệu MDN (mở tab mới)</a>
<a href="mailto:hello@example.com">Gửi email</a>
```

Thực hành từng bước

1. Tìm và sửa tất cả “xem thêm” vô nghĩa thành nội dung có nghĩa.
2. Tạo mailto và tel kiểm tra trên thiết bị hỗ trợ.
3. Đánh giá khi nào cần target _blank thay vì luôn mở tab mới.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “Nội dung liên kết, mở tab mới và an toàn”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Mọi liên kết có đích rõ, nhãn dễ hiểu, không nhầm button. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<a href="https://developer.mozilla.org/" target="_blank" rel="noopener noreferrer">Đọc tài
liệu MDN (mở tab mới)</a>
<a href="mailto:hello@example.com">Gửi email</a>
```

Phân biệt link và button, tránh link gây hiểu lầm.

Checklist

[ ] Đã đọc và giải thích được các khái niệm: Descriptive Link Text, target, rel, mailto, tel, download
[ ] Đã làm: Tìm và sửa tất cả “xem thêm” vô nghĩa thành nội dung có nghĩa.
[ ] Đã làm: Tạo mailto và tel kiểm tra trên thiết bị hỗ trợ.
[ ] Đã làm: Đánh giá khi nào cần target _blank thay vì luôn mở tab mới.
[ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
[ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
[ ] Hoàn thành: Mọi liên kết có đích rõ, nhãn dễ hiểu, không nhầm button.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Structuring_content

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

Anchor, URL tương đối và tuyệt đối

---

3. Menu, breadcrumb và skip link

Tổ chức vùng điều hướng với nav và cấu trúc danh sách.

Mục tiêu học tập

- Hiểu và giải thích được nav trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề “Menu, breadcrumb và skip link” mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi

- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

nav · aria-label · Breadcrumb · Skip Link · main

Giải thích chi tiết

Vì sao cần học trong thực tế?

Tổ chức vùng điều hướng với nav và cấu trúc danh sách. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- nav xác định vùng điều hướng quan trọng; có thể có nhiều nav với nhãn khác nhau.
- Mục menu thường là danh sách liên kết, không cần JavaScript nếu chỉ dẫn trang.
- Skip link giúp người dùng bàn phím bỏ qua menu tới nội dung chính.
- Đường dẫn breadcrumb phản ánh vị trí hiện tại và có thể dùng aria-current="page".

Ví dụ HTML

```
<a href="#main-content">Bỏ qua tới nội dung chính</a>
<nav aria-label="Điều hướng chính"><ul><li><a href="/">Trang chủ</a></li></ul></nav>
<main id="main-content"><h1>Bài học</h1></main>
```

Thực hành từng bước

1. Tạo nav chính và breadcrumb cho trang bài viết.
2. Thêm liên kết bỏ qua tới #main-content.
3. Dùng Tab/Enter để kiểm tra menu không cần chuột.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “Menu, breadcrumb và skip link”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Điều hướng được toàn bộ site chỉ bằng bàn phím. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<a href="#main-content">Bỏ qua tới nội dung chính</a>
<nav aria-label="Điều hướng chính"><ul><li><a href="/">Trang chủ</a></li></ul></nav>
<main id="main-content"><h1>Bài học</h1></main>
```

Tổ chức vùng điều hướng với nav và cấu trúc danh sách.

Checklist

- [ ] Đã đọc và giải thích được các khái niệm: nav, aria-label, Breadcrumb, Skip Link, main
- [ ] Đã làm: Tạo nav chính và breadcrumb cho trang bài viết.
- [ ] Đã làm: Thêm liên kết bỏ qua tới #main-content.
- [ ] Đã làm: Dùng Tab/Enter để kiểm tra menu không cần chuột.
- [ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
- [ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
- [ ] Hoàn thành: Điều hướng được toàn bộ site chỉ bằng bàn phím.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Accessibility/HTML

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

Nội dung liên kết, mở tab mới và an toàn

---

4. Mini project: Website thông tin ba trang

Ứng dụng liên kết, URL, menu và nội dung có cấu trúc.

Mục tiêu học tập

- Hiểu và giải thích được Multi-page Site trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề “Mini project: Website thông tin ba trang” mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

Multi-page Site · Navigation · Relative Paths · Link Audit

Giải thích chi tiết

Vì sao cần học trong thực tế?

Ứng dụng liên kết, URL, menu và nội dung có cấu trúc. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- Mỗi trang có title/h1 riêng và điều hướng nhất quán.

- Không dùng JavaScript để điều hướng khi anchor là đủ.
- Kiểm tra cả link nội bộ, link ngoài, link fragment và lỗi ảnh.
- Tổ chức assets theo đường dẫn có thể hoạt động khi triển khai lên hosting.

Ví dụ HTML

```
<!-- index.html -> about.html -> contact.html
     Kiểm tra: title, h1, nav, links, fragment. -->
```

Thực hành từng bước

1. Làm trang chủ, giới thiệu, liên hệ, tài liệu.
2. Liên kết đến từng phần trong bài viết.
3. Dùng DevTools Network tìm 404 rồi sửa.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “Mini project: Website thông tin ba trang”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Website có ít nhất 3 trang, tối thiểu 12 liên kết hợp lệ. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm


```
<!-- index.html -> about.html -> contact.html
     Kiểm tra: title, h1, nav, links, fragment. -->
```


Ứng dụng liên kết, URL, menu và nội dung có cấu trúc.

Checklist

[ ] Đã đọc và giải thích được các khái niệm: Multi-page Site, Navigation, Relative Paths, Link Audit
[ ] Đã làm: Làm trang chủ, giới thiệu, liên hệ, tài liệu.
[ ] Đã làm: Liên kết đến từng phần trong bài viết.
[ ] Đã làm: Dùng DevTools Network tìm 404 rồi sửa.
[ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp

- [ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
- [ ] Hoàn thành: Website có ít nhất 3 trang, tối thiểu 12 liên kết hợp lệ.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Structuring_content

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

Menu, breadcrumb và skip link

GIAI ĐOẠN 5

05 · Ảnh, đa phương tiện và nội dung nhúng

Làm chủ ngữ nghĩa ảnh, media, hiệu năng và nhúng nội dung đáng tin cậy.

1. img, alt, width, height và lazy loading

Đặt ảnh đúng vai trò và không gây khó tiếp cận.

Mục tiêu học tập

- Hiểu và giải thích được img trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề “img, alt, width, height và lazy loading” mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

img · src · alt · width · height · loading · decoding

Giải thích chi tiết

Vì sao cần học trong thực tế?

Đặt ảnh đúng vai trò và không gây khó tiếp cận. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- alt diễn tả ý nghĩa ảnh trong ngữ cảnh, không máy móc kể mọi chi tiết.
- Ảnh trang trí có thể dùng alt=""; ảnh truyền tải thông tin cần mô tả phù hợp.
- width/height giúp trình duyệt dự trữ tỉ lệ và hạn chế layout shift.
- loading="lazy" hợp lý cho ảnh ngoài màn hình, tránh lạm dụng cho ảnh chính đầu trang.
- Tên file không thay thế alt; CSS background không thay cho ảnh nội dung.

Ví dụ HTML

```
<img src="/assets/images/workshop.webp"
     alt="Sinh viên đang thực hành tại buổi học lập trình"
     width="960" height="640" loading="lazy">
```

Thực hành từng bước

1. Tạo gallery 6 ảnh với alt theo mục đích.
2. Cho một ảnh trang trí alt rỗng và giải thích.
3. Đặt width/height và so sánh layout khi tải chậm.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài "img, alt, width, height và lazy loading". Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Mỗi ảnh đều có quyết định alt có chủ ý; ảnh không méo. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<img src="/assets/images/workshop.webp"
     alt="Sinh viên đang thực hành tại buổi học lập trình"
     width="960" height="640" loading="lazy">
```

Đặt ảnh đúng vai trò và không gây khó tiếp cận.

Checklist

- [ ] Đã đọc và giải thích được các khái niệm: img, src, alt, width, height, loading, decoding
- [ ] Đã làm: Tạo gallery 6 ảnh với alt theo mục đích.
- [ ] Đã làm: Cho một ảnh trang trí alt rỗng và giải thích.
- [ ] Đã làm: Đặt width/height và so sánh layout khi tải chậm.
- [ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
- [ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
- [ ] Hoàn thành: Mỗi ảnh đều có quyết định alt có chủ ý; ảnh không méo.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML/Element/img

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

Mini project: Website thông tin ba trang

2. picture, srcset và sizes

Cung cấp ảnh phù hợp màn hình và định dạng.

Mục tiêu học tập

- Hiểu và giải thích được picture trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "picture, srcset và sizes" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

picture · source · srcset · sizes · WebP · AVIF

Giải thích chi tiết

Vì sao cần học trong thực tế?

Cung cấp ảnh phù hợp màn hình và định dạng. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- srcset cho phép trình duyệt chọn tài nguyên theo mật độ hoặc chiều rộng.
- sizes mô tả kích thước hiển thị dự kiến để chọn đúng ảnh theo w descriptors.
- picture hỗ trợ art direction hoặc định dạng ưu tiên với source và img fallback.
- Không dùng srcset để tự giải quyết CSS layout; phải khai báo kích thước đúng.

Ví dụ HTML

```
<picture>
  <source type="image/webp" srcset="/images/team.webp">
  <img src="/images/team.jpg" width="900" height="600" alt="Nhóm học HTML">
</picture>
```

Thực hành từng bước

1. Tạo ảnh 480w, 960w, 1440w và srcset.
2. Thử đổi viewport trong DevTools và quan sát URL ảnh đã tải.
3. Tạo picture có WebP và JPEG fallback.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài "picture, srcset và sizes". Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Giải thích được sự khác nhau giữa srcset và picture. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<picture>
  <source type="image/webp" srcset="/images/team.webp">
  <img src="/images/team.jpg" width="900" height="600" alt="Nhóm học HTML">
</picture>
```

Cung cấp ảnh phù hợp màn hình và định dạng.

Checklist

[ ] Đã đọc và giải thích được các khái niệm: picture, source, srcset, sizes, WebP, AVIF
[ ] Đã làm: Tạo ảnh 480w, 960w, 1440w và srcset.
[ ] Đã làm: Thử đổi viewport trong DevTools và quan sát URL ảnh đã tải.
[ ] Đã làm: Tạo picture có WebP và JPEG fallback.
[ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
[ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
[ ] Hoàn thành: Giải thích được sự khác nhau giữa srcset và picture.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML/Responsive_images

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

img, alt, width, height và lazy loading

---

3. figure, figcaption, audio và video

Gắn caption và cung cấp điều khiển media.

Mục tiêu học tập

- Hiểu và giải thích được figure trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề “figure, figcaption, audio và video” mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

figure · figcaption · audio · video · source · track · controls

Giải thích chi tiết

Vì sao cần học trong thực tế?

Gắn caption và cung cấp điều khiển media. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- figure/figcaption gắn hình, sơ đồ, mã minh họa với chú thích có nghĩa.
- audio/video có controls giúp người dùng điều khiển mặc định.
- track kind="captions" cung cấp phụ đề cho video; không thay bằng văn bản in trên video.
- poster hiển thị ảnh đại diện video; preload là gợi ý cho trình duyệt.
- Không tự phát video có âm thanh đột ngột.

Ví dụ HTML

```
<figure>
  <img src="/images/chart.png" alt="Biểu đồ tăng trưởng theo năm" width="640" height="400">
  <figcaption>Hình 1: Kết quả khảo sát.</figcaption>
</figure>
<video controls width="640" poster="/images/poster.jpg">
  <source src="/media/lesson.mp4" type="video/mp4">
  <track kind="captions" src="/media/vi.vtt" srclang="vi" label="Tiếng Việt">
</video>
```

Thực hành từng bước

1. Làm một figure có ảnh và caption.
2. Chèn video với controls, poster và file phụ đề mẫu VTT.
3. Tắt tiếng và kiểm tra người dùng có thể hiểu video nhờ phụ đề không.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “figure, figcaption, audio và video”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Video có controls và caption hợp lệ, ảnh có chú thích phù hợp. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<figure>
  <img src="/images/chart.png" alt="Biểu đồ tăng trưởng theo năm" width="640" height="400">
  <figcaption>Hình 1: Kết quả khảo sát.</figcaption>
</figure>
<video controls width="640" poster="/images/poster.jpg">
  <source src="/media/lesson.mp4" type="video/mp4">
  <track kind="captions" src="/media/vi.vtt" srclang="vi" label="Tiếng Việt">
</video>
```

Gắn caption và cung cấp điều khiển media.

Checklist

- [ ] Đã đọc và giải thích được các khái niệm: figure, figcaption, audio, video, source, track, controls
- [ ] Đã làm: Làm một figure có ảnh và caption.
- [ ] Đã làm: Chèn video với controls, poster và file phụ đề mẫu VTT.
- [ ] Đã làm: Tắt tiếng và kiểm tra người dùng có thể hiểu video nhờ phụ đề không.
- [ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
- [ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
- [ ] Hoàn thành: Video có controls và caption hợp lệ, ảnh có chú thích phù hợp.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML/Element/video

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

picture, srcset và sizes

---

4. iframe, nhúng nội dung và quyền hạn

Hiểu khi nào nên nhúng và giới hạn rủi ro.

Mục tiêu học tập

- Hiểu và giải thích được iframe trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề “iframe, nhúng nội dung và quyền hạn” mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

iframe · title · loading · sandbox · allow · Cross-origin

Giải thích chi tiết

Vì sao cần học trong thực tế?

Hiểu khi nào nên nhúng và giới hạn rủi ro. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- iframe hiển thị tài liệu khác trong một vùng; không phải cách thay thế tất cả component.
- Đặt title mô tả cho iframe để công cụ hỗ trợ phân biệt nội dung.
- sandbox hạn chế quyền của tài liệu nhúng nhưng cấu hình sai có thể làm tính năng không hoạt động.
- allow cho phép một số browser features theo chính sách; quyền còn phụ thuộc hệ thống khác.
- Chỉ nhúng nguồn đáng tin và cân nhắc quyền riêng tư của người dùng.

Ví dụ HTML

```
<iframe title="Bản đồ địa điểm học"
        src="https://www.openstreetmap.org/export/embed.html"
        loading="lazy" referrerpolicy="no-referrer"
        width="600" height="400"></iframe>
```

Thực hành từng bước

1. Nhúng bản đồ/video minh họa từ nguồn cho phép.
2. Đặt title, loading=lazy, thử sandbox cơ bản.
3. Đọc yêu cầu quyền nhúng và so sánh trước/sau sandbox.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “iframe, nhúng nội dung và quyền hạn”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.
- Không nhúng trang đăng nhập hoặc nội dung không rõ nguồn gốc để thu thập thông tin.

Tiêu chí hoàn thành

Mô tả được ranh giới bảo mật cơ bản của iframe. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<iframe title="Bản đồ địa điểm học"
        src="https://www.openstreetmap.org/export/embed.html"
        loading="lazy" referrerpolicy="no-referrer"
        width="600" height="400"></iframe>
```

Hiểu khi nào nên nhúng và giới hạn rủi ro.

Checklist

- [ ] Đã đọc và giải thích được các khái niệm: iframe, title, loading, sandbox, allow, Cross-origin
- [ ] Đã làm: Nhúng bản đồ/video minh họa từ nguồn cho phép.
- [ ] Đã làm: Đặt title, loading=lazy, thử sandbox cơ bản.
- [ ] Đã làm: Đọc yêu cầu quyền nhúng và so sánh trước/sau sandbox.
- [ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
- [ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
- [ ] Hoàn thành: Mô tả được ranh giới bảo mật cơ bản của iframe.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML/Element/iframe

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

figure, figcaption, audio và video

GIAI ĐOẠN 6

06 · Bảng dữ liệu và biểu mẫu HTML

Xây bảng có ý nghĩa và biểu mẫu gửi dữ liệu rõ ràng, dễ dùng và an toàn.

1. Table chỉ dành cho dữ liệu dạng bảng

Sử dụng table, caption, thead, tbody, th, td và scope.

Mục tiêu học tập

- Hiểu và giải thích được table trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "Table chỉ dành cho dữ liệu dạng bảng" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

table · caption · thead · tbody · tr · th · td · scope

Giải thích chi tiết

Vì sao cần học trong thực tế?

Sử dụng table, caption, thead, tbody, th, td và scope. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- table dành cho dữ liệu có quan hệ hàng/cột; không dùng làm layout toàn trang.
- caption đặt tên bảng; th biểu thị ô tiêu đề; scope row/col giúp xác định quan hệ.
- thead/tbody/tfoot tổ chức các nhóm hàng.
- colspan/rowspan dùng khi dữ liệu thật sự cần gộp ô, không lạm dụng.
- Bảng phức tạp có thể cần headers/id hoặc cấu trúc lại cho dễ đọc.

Ví dụ HTML

```
<table>
  <caption>Lịch học trong tuần</caption>
  <thead><tr><th scope="col">Ngày</th><th scope="col">Chủ đề</th></tr></thead>
  <tbody><tr><th scope="row">Thứ hai</th><td>HTML cơ bản</td></tr></tbody>
</table>
```

Thực hành từng bước

1. Tạo bảng lịch học có tiêu đề hàng và cột.
2. Đặt caption, scope=col và scope=row.
3. Dùng công cụ đọc bảng/keyboard kiểm tra ý nghĩa.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài "Table chỉ dành cho dữ liệu dạng bảng". Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Bảng đọc được theo hàng/cột bằng trình đọc màn hình. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<table>
  <caption>Lịch học trong tuần</caption>
  <thead><tr><th scope="col">Ngày</th><th scope="col">Chủ đề</th></tr></thead>
  <tbody><tr><th scope="row">Thứ hai</th><td>HTML cơ bản</td></tr></tbody>
</table>
```

Sử dụng table, caption, thead, tbody, th, td và scope.

Checklist

[ ] Đã đọc và giải thích được các khái niệm: table, caption, thead, tbody, tr, th, td, scope

[ ] Đã làm: Tạo bảng lịch học có tiêu đề hàng và cột.

[ ] Đã làm: Đặt caption, scope=col và scope=row.

[ ] Đã làm: Dùng công cụ đọc bảng/keyboard kiểm tra ý nghĩa.

[ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp

[ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc

[ ] Hoàn thành: Bảng đọc được theo hàng/cột bằng trình đọc màn hình.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Structuring_content/HTML_table_basics

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

iframe, nhúng nội dung và quyền hạn

2. form, action, method, name và submit

Hiểu dữ liệu nào thực sự được gửi và gửi tới đâu.

Mục tiêu học tập

- Hiểu và giải thích được form trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "form, action, method, name và submit" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

form · action · method · name · value · GET · POST

Giải thích chi tiết

Vì sao cần học trong thực tế?

Hiểu dữ liệu nào thực sự được gửi và gửi tới đâu. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- action là đích nhận dữ liệu; method GET hay POST ảnh hưởng cách chuyển dữ liệu.
- GET đặt thông số trong URL, phù hợp tìm kiếm không nhạy cảm; POST gửi dữ liệu trong request body, không đồng nghĩa mã hóa.
- Chỉ các control đủ điều kiện và có name mới tạo cặp tên–giá trị khi submit.
- button type="submit" gửi biểu mẫu; button type="button" không submit mặc định.
- HTML không xử lý dữ liệu trên máy chủ; phải có backend hoặc endpoint.

Ví dụ HTML

```
<form action="/search" method="get">
  <label for="q">Từ khóa</label>
  <input id="q" name="q" type="search" required>
  <button type="submit">Tìm kiếm</button>
</form>
```

Thực hành từng bước

1. Tạo form tìm kiếm method=get, gửi tới endpoint thử nghiệm.
2. Quan sát query string và tên trường.
3. Đổi thử POST để hiểu request body bằng Network.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “form, action, method, name và submit”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.
- Không gửi mật khẩu hoặc dữ liệu cá nhân nhạy cảm bằng GET; POST cũng cần HTTPS và kiểm tra phía server.

Tiêu chí hoàn thành

Giải thích được vì sao bỏ name khiến giá trị trường không được gửi. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<form action="/search" method="get">
  <label for="q">Từ khóa</label>
  <input id="q" name="q" type="search" required>
  <button type="submit">Tìm kiếm</button>
</form>
```

Hiểu dữ liệu nào thực sự được gửi và gửi tới đâu.

Checklist

[ ] Đã đọc và giải thích được các khái niệm: form, action, method, name, value, GET, POST

[ ] Đã làm: Tạo form tìm kiếm method=get, gửi tới endpoint thử nghiệm.

[ ] Đã làm: Quan sát query string và tên trường.

[ ] Đã làm: Đổi thử POST để hiểu request body bằng Network.

[ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp

[ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc

[ ] Hoàn thành: Giải thích được vì sao bỏ name khiến giá trị trường không được gửi.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Structuring_content/HTML_forms

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

Table chỉ dành cho dữ liệu dạng bảng

3. label và các kiểu input phổ biến

Chọn type chính xác và liên kết nhãn với control.

Mục tiêu học tập

- Hiểu và giải thích được label trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "label và các kiểu input phổ biến" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

label · input type · id · name · autocomplete · placeholder

Giải thích chi tiết

Vì sao cần học trong thực tế?

Chọn type chính xác và liên kết nhãn với control. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- label for cần trùng id của control, click vào label sẽ tập trung control.
- input type=email, url, number, date, tel, search, password, checkbox, radio mang hành vi riêng.
- placeholder là gợi ý ngắn, không thay thế label.
- autocomplete hỗ trợ nhập liệu nhưng phải dùng token đúng mục đích.
- Radio cùng name tạo nhóm chọn một; checkbox cho phép chọn nhiều.

Ví dụ HTML

```
<label for="email">Email</label>
<input id="email" name="email" type="email" autocomplete="email" required>
<fieldset>
  <legend>Nhận thông báo?</legend>
  <label><input type="radio" name="notify" value="yes"> Có</label>
  <label><input type="radio" name="notify" value="no"> Không</label>
</fieldset>
```

Thực hành từng bước

1. Tạo form đăng ký có email, mật khẩu, ngày sinh, checkbox và radio.
2. Tắt CSS để kiểm tra form vẫn có nhãn.
3. Dùng bàn phím chọn radio/checkbox và tab qua các trường.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “label và các kiểu input phổ biến”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Form có nhãn đầy đủ và loại input phù hợp dữ liệu. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<label for="email">Email</label>
<input id="email" name="email" type="email" autocomplete="email" required>
<fieldset>
  <legend>Nhận thông báo?</legend>
  <label><input type="radio" name="notify" value="yes"> Có</label>
  <label><input type="radio" name="notify" value="no"> Không</label>
</fieldset>
```

Chọn type chính xác và liên kết nhãn với control.

Checklist

[ ] Đã đọc và giải thích được các khái niệm: label, input type, id, name, autocomplete, placeholder

[ ] Đã làm: Tạo form đăng ký có email, mật khẩu, ngày sinh, checkbox và radio.

[ ] Đã làm: Tắt CSS để kiểm tra form vẫn có nhãn.

[ ] Đã làm: Dùng bàn phím chọn radio/checkbox và tab qua các trường.

[ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp

[ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc

[ ] Hoàn thành: Form có nhãn đầy đủ và loại input phù hợp dữ liệu.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML/Element/input

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

form, action, method, name và submit

4. select, textarea, fieldset, datalist và output

Tổ chức biểu mẫu dài dễ hiểu.

Mục tiêu học tập

- Hiểu và giải thích được textarea trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề “select, textarea, fieldset, datalist và output” mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

textarea · select · option · optgroup · fieldset · legend · datalist · output

Giải thích chi tiết

Vì sao cần học trong thực tế?

Tổ chức biểu mẫu dài dễ hiểu. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- select phù hợp danh sách lựa chọn đã xác định; textarea cho văn bản nhiều dòng.
- fieldset và legend nhóm các control có chung câu hỏi.
- datalist đưa gợi ý nhưng không buộc giá trị thuộc danh sách.
- output hiển thị kết quả tính toán nếu có logic cập nhật; tự HTML không tự tính toán.
- Không ẩn nhãn chỉ để form đẹp; CSS có thể làm đẹp sau.

Ví dụ HTML

```
<label for="topic">Chủ đề quan tâm</label>
<select id="topic" name="topic">
  <option value="html">HTML</option>
  <option value="accessibility">Accessibility</option>
</select>
<label for="feedback">Góp ý</label>
<textarea id="feedback" name="feedback" rows="4"></textarea>
```

Thực hành từng bước

1. Tạo form khảo sát có nhóm fieldset và dropdown.
2. Tạo textarea có maxlength, rows.
3. So sánh select với datalist khi người dùng nhập giá trị tự do.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “select, textarea, fieldset, datalist và output”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Phân nhóm form dài thành các phần mạch lạc. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<label for="topic">Chủ đề quan tâm</label>
<select id="topic" name="topic">
  <option value="html">HTML</option>
  <option value="accessibility">Accessibility</option>
</select>
<label for="feedback">Góp ý</label>
<textarea id="feedback" name="feedback" rows="4"></textarea>
```

Tổ chức biểu mẫu dài dễ hiểu.

Checklist

- [ ] Đã đọc và giải thích được các khái niệm: textarea, select, option, optgroup, fieldset, legend, datalist, output
- [ ] Đã làm: Tạo form khảo sát có nhóm fieldset và dropdown.
- [ ] Đã làm: Tạo textarea có maxlength, rows.
- [ ] Đã làm: So sánh select với datalist khi người dùng nhập giá trị tự do.
- [ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
- [ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
- [ ] Hoàn thành: Phân nhóm form dài thành các phần mạch lạc.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Learn_web_development/Extensions/Forms/How_to_structure_a_web_form

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

label và các kiểu input phổ biến

5. Native validation và giới hạn của kiểm tra phía client

Sử dụng required, min/max, minlength, pattern mà không ảo tưởng bảo mật.

Mục tiêu học tập

- Hiểu và giải thích được required trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "Native validation và giới hạn của kiểm tra phía client" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

required · minlength · maxlength · min · max · pattern · Validity

Giải thích chi tiết

Vì sao cần học trong thực tế?

Sử dụng required, min/max, minlength, pattern mà không ảo tưởng bảo mật. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- required, minlength, maxlength, min/max, pattern hỗ trợ kiểm tra ngay trong trình duyệt.
- type=email kiểm tra cú pháp dạng email cơ bản, không chứng minh email tồn tại.
- pattern là biểu thức chính quy với quy tắc riêng của HTML; cần thử ví dụ hợp lệ/không hợp lệ.
- novalidate bỏ kiểm tra ràng buộc tương tác khi submit form.
- Client-side validation tăng UX, không thể thay thế xác thực và kiểm tra dữ liệu phía server.

Ví dụ HTML

```
<form action="/register" method="post">
  <label for="username">Tên người dùng (3–20 ký tự)</label>
  <input id="username" name="username" required minlength="3" maxlength="20" pattern="[A-Za-
z0-9_]+">
  <button>Đăng ký</button>
</form>
```

Thực hành từng bước

1. Tạo trường username 3–20 ký tự với pattern phù hợp.
2. Thử gửi form thiếu hoặc sai dữ liệu và quan sát thông báo.
3. Gửi request thủ công không qua form để hiểu server phải kiểm tra lại.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “Native validation và giới hạn của kiểm tra phía client”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.
- Đừng đưa quy tắc nhạy cảm hoặc coi pattern HTML là cơ chế chống tấn công.

Tiêu chí hoàn thành

Liệt kê ít nhất 4 trường hợp server phải tự xác thực lại. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<form action="/register" method="post">
  <label for="username">Tên người dùng (3–20 ký tự)</label>
  <input id="username" name="username" required minlength="3" maxlength="20" pattern="[A-Za-
z0-9_]+">
  <button>Đăng ký</button>
</form>
```

Sử dụng required, min/max, minlength, pattern mà không ảo tưởng bảo mật.

Checklist

[ ] Đã đọc và giải thích được các khái niệm: required, minlength, maxlength, min, max, pattern, Validity

[ ] Đã làm: Tạo trường username 3–20 ký tự với pattern phù hợp.

[ ] Đã làm: Thử gửi form thiếu hoặc sai dữ liệu và quan sát thông báo.

[ ] Đã làm: Gửi request thủ công không qua form để hiểu server phải kiểm tra lại.

[ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp

[ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc

[ ] Hoàn thành: Liệt kê ít nhất 4 trường hợp server phải tự xác thực lại.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Learn_web_development/Extensions/Forms/Form_validation

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

select, textarea, fieldset, datalist và output

6. File upload, enctype và giới hạn an toàn

Hiểu multipart/form-data và file input.

Mục tiêu học tập

- Hiểu và giải thích được file trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề “File upload, enctype và giới hạn an toàn” mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

file · accept · multiple · enctype · multipart/form-data

Giải thích chi tiết

Vì sao cần học trong thực tế?

Hiểu multipart/form-data và file input. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- input type=file cho người dùng chọn file, không cho website tùy ý đọc ổ đĩa.
- POST với enctype="multipart/form-data" là cấu hình phổ biến để upload file.
- accept chỉ gợi ý loại file cho file picker, không bảo đảm an toàn hoặc loại thật.
- multiple cho phép chọn nhiều file; server vẫn phải giới hạn dung lượng và xác minh nội dung.
- Không hiển thị công khai file upload thiếu kiểm tra quyền và định dạng.

Ví dụ HTML

```
<form action="/upload" method="post" enctype="multipart/form-data">
  <label for="avatar">Ảnh đại diện</label>
  <input id="avatar" name="avatar" type="file" accept="image/*">
  <button type="submit">Tải lên</button>
</form>
```

Thực hành từng bước

1. Tạo form upload avatar có accept=image/*.
2. Quan sát request trong Network khi có endpoint thử nghiệm.
3. Mô tả checklist kiểm tra server: type, size, filename, quyền, lưu trữ.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “File upload, enctype và giới hạn an toàn”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Giải thích được vì sao accept không phải biện pháp bảo mật. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<form action="/upload" method="post" enctype="multipart/form-data">
  <label for="avatar">Ảnh đại diện</label>
  <input id="avatar" name="avatar" type="file" accept="image/*">
  <button type="submit">Tải lên</button>
</form>
```

Hiểu multipart/form-data và file input.

Checklist

[ ] Đã đọc và giải thích được các khái niệm: file, accept, multiple, enctype, multipart/form-data
[ ] Đã làm: Tạo form upload avatar có accept=image/*.
[ ] Đã làm: Quan sát request trong Network khi có endpoint thử nghiệm.
[ ] Đã làm: Mô tả checklist kiểm tra server: type, size, filename, quyền, lưu trữ.
[ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
[ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
[ ] Hoàn thành: Giải thích được vì sao accept không phải biện pháp bảo mật.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML/Element/input/file

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

Native validation và giới hạn của kiểm tra phía client

---

7. Mini project: Form đăng ký sự kiện có kiểm tra

Hoàn thiện một form đáp ứng tính hữu dụng và tính dễ tiếp cận.

Mục tiêu học tập

- Hiểu và giải thích được Registration Form trong ngữ cảnh HTML thực tế

- Tự triển khai chủ đề “Mini project: Form đăng ký sự kiện có kiểm tra” mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

Registration Form · Validation · Fieldset · Privacy · Server Boundary

Giải thích chi tiết

Vì sao cần học trong thực tế?

Hoàn thiện một form đáp ứng tính hữu dụng và tính dễ tiếp cận. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- Form đăng ký phải có mục đích, thông tin bắt buộc tối thiểu và mô tả cách dùng dữ liệu.
- Trường nhạy cảm cần chỉ yêu cầu khi thực sự cần; không lấy dữ liệu thừa.
- Label, error message và thứ tự keyboard quan trọng hơn hình thức bóng bẩy.
- Chưa có backend thì không tuyên bố dữ liệu đã được lưu/gửi thành công.

Ví dụ HTML

```
<form action="/api/register" method="post">
  <label for="full-name">Họ tên</label>
  <input id="full-name" name="fullName" autocomplete="name" required>
  <label for="reg-email">Email</label>
  <input id="reg-email" name="email" type="email" autocomplete="email" required>
  <button type="submit">Gửi đăng ký</button>
</form>
```

Thực hành từng bước

1. Làm form đăng ký buổi học với họ tên, email, hình thức tham gia, chủ đề và đồng ý chính sách.
2. Kiểm tra trường trống/sai, Tab/Shift+Tab.
3. Mô tả endpoint cần thiết nhưng không giả lập thành công.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “Mini project: Form đăng ký sự kiện có kiểm tra”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Form điều hướng bằng bàn phím, có kiểm tra cơ bản và nêu rõ cần backend. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<form action="/api/register" method="post">
  <label for="full-name">Họ tên</label>
  <input id="full-name" name="fullName" autocomplete="name" required>
  <label for="reg-email">Email</label>
  <input id="reg-email" name="email" type="email" autocomplete="email" required>
  <button type="submit">Gửi đăng ký</button>
</form>
```

Hoàn thiện một form đáp ứng tính hữu dụng và tính dễ tiếp cận.

Checklist

- [ ] Đã đọc và giải thích được các khái niệm: Registration Form, Validation, Fieldset, Privacy, Server Boundary
- [ ] Đã làm: Làm form đăng ký buổi học với họ tên, email, hình thức tham gia, chủ đề và đồng ý chính sách.
- [ ] Đã làm: Kiểm tra trường trống/sai, Tab/Shift+Tab.
- [ ] Đã làm: Mô tả endpoint cần thiết nhưng không giả lập thành công.
- [ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
- [ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
- [ ] Hoàn thành: Form điều hướng bằng bàn phím, có kiểm tra cơ bản và nêu rõ cần backend.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Structuring_content

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

File upload, enctype và giới hạn an toàn

GIAI ĐOẠN 7

07 · Semantic HTML và khả năng tiếp cận

Thay div soup bằng cấu trúc có ý nghĩa và phục vụ người dùng thực tế.

1. header, nav, main, section, article, aside, footer

Xác định vùng và ý nghĩa của nội dung.

Mục tiêu học tập

- Hiểu và giải thích được Landmark trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "header, nav, main, section, article, aside, footer" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

Landmark · header · nav · main · section · article · aside · footer

Giải thích chi tiết

Vì sao cần học trong thực tế?

Xác định vùng và ý nghĩa của nội dung. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- main xác định nội dung chính của tài liệu; thường có một main hiển thị cho nội dung chính.
- article cho nội dung có thể tự đứng độc lập; section nhóm theo chủ đề thường có heading.
- aside chứa nội dung liên quan gián tiếp; header/footer có ngữ cảnh tùy nơi đặt.
- Không thay tất cả div bằng section nếu phần đó không phải nhóm nội dung có chủ đề.
- Semantic HTML giúp công cụ hỗ trợ duyệt vùng trang và người đọc hiểu cấu trúc.

Ví dụ HTML

```
<header><nav aria-label="Chính">...</nav></header>
<main>
  <article><h1>Học HTML</h1><p>...</p></article>
  <aside aria-label="Bài viết liên quan">...</aside>
</main>
<footer>Thông tin liên hệ</footer>
```

Thực hành từng bước

1. Chuyển một trang toàn div sang HTML semantic.
2. Kiểm tra landmarks bằng Accessibility tree.
3. Giải thích tại sao một phần cần article, phần khác chỉ cần div.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài "header, nav, main, section, article, aside, footer". Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Không lạm dụng section/article; mọi landmark có mục đích rõ. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<header><nav aria-label="Chính">...</nav></header>
<main>
  <article><h1>Học HTML</h1><p>...</p></article>
  <aside aria-label="Bài viết liên quan">...</aside>
</main>
<footer>Thông tin liên hệ</footer>
```

Xác định vùng và ý nghĩa của nội dung.

Checklist

[ ] Đã đọc và giải thích được các khái niệm: Landmark, header, nav, main, section, article, aside, footer
[ ] Đã làm: Chuyển một trang toàn div sang HTML semantic.
[ ] Đã làm: Kiểm tra landmarks bằng Accessibility tree.
[ ] Đã làm: Giải thích tại sao một phần cần article, phần khác chỉ cần div.
[ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
[ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
[ ] Hoàn thành: Không lạm dụng section/article; mọi landmark có mục đích rõ.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Glossary/Semantics

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

Mini project: Form đăng ký sự kiện có kiểm tra

---

2. button, details, summary và dialog

Ưu tiên tương tác gốc của trình duyệt thay vì div click.

Mục tiêu học tập

- Hiểu và giải thích được button trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "button, details, summary và dialog" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

button · details · summary · dialog · popover

Giải thích chi tiết

Vì sao cần học trong thực tế?

Ưu tiên tương tác gốc của trình duyệt thay vì div click. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- button có khả năng truy cập bàn phím và ngữ nghĩa hành động mặc định.
- details/summary cung cấp thao tác đóng/mở không cần JavaScript đơn giản.
- dialog cung cấp cấu trúc hộp thoại, nhưng nhiều hành vi mở/đóng cần JavaScript để quản lý đúng.
- popover là cơ chế HTML cho nội dung nổi ở trình duyệt hỗ trợ, cần kiểm tra tương thích.
- Không tạo click handler trên div khi button sẵn có thể làm đúng.

Ví dụ HTML

```
<details>
  <summary>HTML có phải ngôn ngữ lập trình?</summary>
  <p>HTML là ngôn ngữ đánh dấu.</p>
</details>
<button type="button">Mở trợ giúp</button>
```

Thực hành từng bước

1. Tạo FAQ bằng details/summary.
2. Thử Tab, Enter, Space trên button và summary.
3. So sánh với div giả button và ghi các thứ cần tự bổ sung để tương đương.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “button, details, summary và dialog”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Dùng được FAQ không JavaScript và không dùng div giả button. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<details>
  <summary>HTML có phải ngôn ngữ lập trình?</summary>
  <p>HTML là ngôn ngữ đánh dấu.</p>
</details>
<button type="button">Mở trợ giúp</button>
```

Ưu tiên tương tác gốc của trình duyệt thay vì div click.

Checklist

[ ] Đã đọc và giải thích được các khái niệm: button, details, summary, dialog, popover

[ ] Đã làm: Tạo FAQ bằng details/summary.

[ ] Đã làm: Thử Tab, Enter, Space trên button và summary.

[ ] Đã làm: So sánh với div giả button và ghi các thứ cần tự bổ sung để tương đương.

[ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp

[ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc

[ ] Hoàn thành: Dùng được FAQ không JavaScript và không dùng div giả button.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML/Element/details

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

header, nav, main, section, article, aside, footer

3. Accessibility: screen reader, keyboard và focus

Xây tư duy thiết kế cho nhiều phương thức tương tác.

Mục tiêu học tập

- Hiểu và giải thích được Accessibility trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "Accessibility: screen reader, keyboard và focus" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

Accessibility · Screen Reader · Keyboard · Focus · Accessible Name · Alt Text

Giải thích chi tiết

Vì sao cần học trong thực tế?

Xây tư duy thiết kế cho nhiều phương thức tương tác. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- Khả năng tiếp cận bao gồm người dùng bàn phím, trình đọc màn hình và nhiều nhu cầu khác.
- Tên truy cập (accessible name) phải nói rõ chức năng của điều khiển.
- Thứ tự DOM hợp lý quyết định thứ tự đọc và thường ảnh hưởng tab order.
- Focus phải nhìn thấy được; không dùng tabindex dương để vá layout tùy tiện.
- Kiểm thử tự động không thể phát hiện mọi vấn đề ngữ nghĩa hoặc trải nghiệm.

Ví dụ HTML

```
<a href="#content">Bỏ qua điều hướng</a>
<nav aria-label="Chính">...</nav>
<main id="content"><h1>Nội dung</h1></main>
```

Thực hành từng bước

1. Duyệt trang chỉ bằng Tab, Shift+Tab, Enter, Space.
2. Thử VoiceOver/NVDA nếu có điều kiện.
3. Liệt kê 10 lỗi gặp trên một trang mẫu: alt, label, heading, tên link, focus.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “Accessibility: screen reader, keyboard và focus”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Tự duyệt toàn bộ nhiệm vụ chính mà không cần chuột. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<a href="#content">Bỏ qua điều hướng</a>
<nav aria-label="Chính">...</nav>
<main id="content"><h1>Nội dung</h1></main>
```

Xây tư duy thiết kế cho nhiều phương thức tương tác.

Checklist

- [ ] Đã đọc và giải thích được các khái niệm: Accessibility, Screen Reader, Keyboard, Focus, Accessible Name, Alt Text
- [ ] Đã làm: Duyệt trang chỉ bằng Tab, Shift+Tab, Enter, Space.
- [ ] Đã làm: Thử VoiceOver/NVDA nếu có điều kiện.
- [ ] Đã làm: Liệt kê 10 lỗi gặp trên một trang mẫu: alt, label, heading, tên link, focus.
- [ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
- [ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
- [ ] Hoàn thành: Tự duyệt toàn bộ nhiệm vụ chính mà không cần chuột.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Accessibility/HTML

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

button, details, summary và dialog

---

4. ARIA cơ bản và nguyên tắc HTML trước ARIA

Dùng aria-label, aria-describedby, aria-current khi thật cần.

Mục tiêu học tập

- Hiểu và giải thích được ARIA trong ngữ cảnh HTML thực tế

- Tự triển khai chủ đề “ARIA cơ bản và nguyên tắc HTML trước ARIA” mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

ARIA · aria-label · aria-describedby · aria-current · aria-hidden · role

Giải thích chi tiết

Vì sao cần học trong thực tế?

Dùng aria-label, aria-describedby, aria-current khi thật cần. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- Ưu tiên phần tử native; ARIA bổ sung ngữ nghĩa khi thiếu, không tự tạo chức năng.
- aria-label có thể đặt tên cho control không có nhãn hiển thị nhưng nên ưu tiên nhãn thấy được.
- aria-describedby nối mô tả phụ bằng id; aria-current xác định mục hiện tại trong tập hợp.
- aria-hidden="true" loại nội dung khỏi accessibility tree và có thể gây lỗi nếu áp dụng lên phần tử tương tác.
- role không thay hành vi keyboard: role="button" trên div chưa thành button thực sự.

Ví dụ HTML

```
<label for="student-id">Mã học viên</label>
<p id="id-help">Nhập 8 chữ số trên thẻ.</p>
<input id="student-id" name="studentId" aria-describedby="id-help">
```

Thực hành từng bước

1. Tạo input có label và đoạn trợ giúp aria-describedby.
2. Đánh dấu trang hiện tại trong breadcrumb bằng aria-current=page.
3. Cố tình thêm aria-hidden lên vùng chứa button và giải thích rủi ro.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “ARIA cơ bản và nguyên tắc HTML trước ARIA”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.

- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.
- Không dùng ARIA chỉ để làm cho HTML trông chuyên nghiệp.

Tiêu chí hoàn thành

Giải thích quy tắc "native HTML trước ARIA" bằng 3 ví dụ. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<label for="student-id">Mã học viên</label>
<p id="id-help">Nhập 8 chữ số trên thẻ.</p>
<input id="student-id" name="studentId" aria-describedby="id-help">
```

Dùng aria-label, aria-describedby, aria-current khi thật cần.

Checklist

- [ ] Đã đọc và giải thích được các khái niệm: ARIA, aria-label, aria-describedby, aria-current, aria-hidden, role
- [ ] Đã làm: Tạo input có label và đoạn trợ giúp aria-describedby.
- [ ] Đã làm: Đánh dấu trang hiện tại trong breadcrumb bằng aria-current=page.
- [ ] Đã làm: Cố tình thêm aria-hidden lên vùng chứa button và giải thích rủi ro.
- [ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
- [ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
- [ ] Hoàn thành: Giải thích quy tắc "native HTML trước ARIA" bằng 3 ví dụ.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Web/Accessibility/ARIA

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

Accessibility: screen reader, keyboard và focus

---

5. Ngôn ngữ, hướng chữ và nội dung quốc tế

Khai báo lang, dir, bdi, ruby khi phù hợp.

Mục tiêu học tập

- Hiểu và giải thích được lang trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "Ngôn ngữ, hướng chữ và nội dung quốc tế" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

lang · dir · bdi · bdo · Internationalization

Giải thích chi tiết

Vì sao cần học trong thực tế?

Khai báo lang, dir, bdi, ruby khi phù hợp. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- html lang="vi" xác định ngôn ngữ tài liệu; lang cục bộ hữu ích cho đoạn ngoại ngữ.
- dir="rtl" dành cho các ngôn ngữ viết từ phải qua trái trong ngữ cảnh phù hợp.
- bdi cô lập hướng văn bản nhập từ người dùng khi trộn LTR/RTL.
- Dùng UTF-8 và văn bản Unicode; không thay tiếng Việt bằng hình ảnh chữ.
- Không tự suy đoán ngôn ngữ từ địa chỉ IP để gắn sai lang.

Ví dụ HTML

```
<p>Thuật ngữ <span lang="en">accessibility</span> là khả năng tiếp cận.</p>
<p>Tài khoản: <bdi>□□□ >p/<>idb/<□□□□□□□□□
```

Thực hành từng bước

1. Làm trang tiếng Việt xen 2 câu tiếng Anh với lang=en.
2. Thử tên người dùng có chữ RTL trong bdi.
3. Kiểm tra title, alt, label có cùng ngôn ngữ phù hợp.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “Ngôn ngữ, hướng chữ và nội dung quốc tế”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Trình đọc màn hình có thông tin ngôn ngữ phù hợp. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<p>Thuật ngữ <span lang="en">accessibility</span> là khả năng tiếp cận.</p>
<p>Tài khoản: <bdi>□□□ >p/<>idb/<□□□□□□□□□
```

Khai báo lang, dir, bdi, ruby khi phù hợp.

Checklist

- [ ] Đã đọc và giải thích được các khái niệm: lang, dir, bdi, bdo, Internationalization
- [ ] Đã làm: Làm trang tiếng Việt xen 2 câu tiếng Anh với lang=en.
- [ ] Đã làm: Thử tên người dùng có chữ RTL trong bdi.
- [ ] Đã làm: Kiểm tra title, alt, label có cùng ngôn ngữ phù hợp.
- [ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
- [ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
- [ ] Hoàn thành: Trình đọc màn hình có thông tin ngôn ngữ phù hợp.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML/Global_attributes/lang

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

ARIA cơ bản và nguyên tắc HTML trước ARIA

---

6. Mini project: Audit và sửa trang HTML kém chất lượng

Đo chất lượng bằng việc thực sự tìm và sửa lỗi.

Mục tiêu học tập

- Hiểu và giải thích được Accessibility Audit trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề “Mini project: Audit và sửa trang HTML kém chất lượng” mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

Accessibility Audit · Semantic Refactor · Manual Testing · WCAG

Giải thích chi tiết

Vì sao cần học trong thực tế?

Đo chất lượng bằng việc thực sự tìm và sửa lỗi. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- Một điểm số Lighthouse tốt không chứng minh trang hoàn toàn accessible.
- Tiêu chí thực tế: hoàn tất hành động bằng bàn phím, hiểu nội dung không CSS, nhãn có nghĩa.

- Ghi lỗi theo mức tác động, cách tái hiện và thay đổi đã sửa.
- Sử dụng WCAG như chuẩn tham khảo, không tuyên bố chứng nhận khi chưa đánh giá đầy đủ.

Ví dụ HTML

```
<!-- Trước: <div onclick="...">Gửi</div>
     Sau:   <button type="submit">Gửi biểu mẫu</button> -->
```

Thực hành từng bước

1. Tạo bản lỗi cố ý gồm div button, image không alt, heading nhảy cấp và input thiếu label.
2. Liệt kê từng vấn đề, sửa và chụp kết quả kiểm thử.
3. Viết báo cáo trước/sau khoảng 1 trang.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “Mini project: Audit và sửa trang HTML kém chất lượng”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Sửa được ít nhất 8 vấn đề và có ghi chép kiểm thử trước/sau. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm


```
<!-- Trước: <div onclick="...">Gửi</div>
     Sau:   <button type="submit">Gửi biểu mẫu</button> -->
```


Đo chất lượng bằng việc thực sự tìm và sửa lỗi.

Checklist

[ ] Đã đọc và giải thích được các khái niệm: Accessibility Audit, Semantic Refactor, Manual Testing, WCAG
[ ] Đã làm: Tạo bản lỗi cố ý gồm div button, image không alt, heading nhảy cấp và input thiếu label.
[ ] Đã làm: Liệt kê từng vấn đề, sửa và chụp kết quả kiểm thử.
[ ] Đã làm: Viết báo cáo trước/sau khoảng 1 trang.
[ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp

- [ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
- [ ] Hoàn thành: Sửa được ít nhất 8 vấn đề và có ghi chép kiểm thử trước/sau.

Tài liệu tham khảo

- Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
  https://www.w3.org/WAI/tutorials/
- MDN HTML Reference Tài liệu
  https://developer.mozilla.org/en-US/docs/Web/HTML
- web.dev Learn HTML Khóa học
  https://web.dev/learn/html
- Nu HTML Checker Công cụ
  https://validator.w3.org/

Nên học trước

Ngôn ngữ, hướng chữ và nội dung quốc tế

GIAI ĐOẠN 8

08 · SEO, hiệu năng và an toàn nền tảng

Kết nối HTML với tìm kiếm, tốc độ tải, chính sách bảo mật và dữ liệu người dùng.

1. SEO on-page: title, headings, link và mô tả

Hiểu HTML giúp công cụ tìm kiếm đọc nội dung.

Mục tiêu học tập

- Hiểu và giải thích được SEO trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "SEO on-page: title, headings, link và mô tả" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

SEO · title · meta description · heading · canonical · anchor

Giải thích chi tiết

Vì sao cần học trong thực tế?

Hiểu HTML giúp công cụ tìm kiếm đọc nội dung. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- Nội dung hữu ích và cấu trúc dễ đọc quan trọng hơn nhồi từ khóa.
- title nên độc nhất, mô tả đúng trang; meta description là gợi ý trích đoạn, không đảm bảo hiển thị.
- Đường dẫn liên kết và anchor text cung cấp ngữ cảnh cho trang đích.
- Canonical có ý nghĩa trong trường hợp URL trùng lặp và phải khớp ý định xuất bản.
- meta keywords không phải yếu tố xếp hạng phổ biến cần tối ưu.

Ví dụ HTML

```
<head>
  <title>Hướng dẫn HTML cho người mới | HTML Lab</title>
  <meta name="description" content="Học HTML từ cấu trúc cơ bản đến biểu mẫu và
accessibility qua ví dụ thực hành.">
</head>
```

Thực hành từng bước

1. Viết title/description cho 3 trang có mục đích khác nhau.
2. Rà h1/h2 và internal links.
3. Dùng Google Search Central đối chiếu nguyên tắc chính thức.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài "SEO on-page: title, headings, link và mô tả". Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Mỗi trang có metadata riêng và nội dung đúng với lời hứa trên title. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<head>
  <title>Hướng dẫn HTML cho người mới | HTML Lab</title>
  <meta name="description" content="Học HTML từ cấu trúc cơ bản đến biểu mẫu và
accessibility qua ví dụ thực hành.">
</head>
```

Hiểu HTML giúp công cụ tìm kiếm đọc nội dung.

Checklist

[ ] Đã đọc và giải thích được các khái niệm: SEO, title, meta description, heading, canonical, anchor

[ ] Đã làm: Viết title/description cho 3 trang có mục đích khác nhau.

[ ] Đã làm: Rà h1/h2 và internal links.

[ ] Đã làm: Dùng Google Search Central đối chiếu nguyên tắc chính thức.

[ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp

[ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc

[ ] Hoàn thành: Mỗi trang có metadata riêng và nội dung đúng với lời hứa trên title.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developers.google.com/search/docs/fundamentals/seo-starter-guide

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

Mini project: Audit và sửa trang HTML kém chất lượng

2. Structured data và JSON-LD nhập môn

Phân biệt dữ liệu có cấu trúc với thẻ HTML ngữ nghĩa.

Mục tiêu học tập

- Hiểu và giải thích được Structured Data trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "Structured data và JSON-LD nhập môn" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

Structured Data · JSON-LD · schema.org · Rich Result

Giải thích chi tiết

Vì sao cần học trong thực tế?

Phân biệt dữ liệu có cấu trúc với thẻ HTML ngữ nghĩa. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- JSON-LD là một cách khai báo dữ liệu có cấu trúc; thường đặt trong script type="application/ld+json".
- Chỉ mô tả nội dung thực sự có trên trang, không bịa đánh giá hoặc thông tin sản phẩm.
- Dữ liệu có cấu trúc không đảm bảo kết quả tìm kiếm phong phú.
- Nên dùng loại schema phù hợp và kiểm tra bằng công cụ được hỗ trợ.
- Không nhầm JSON-LD với JavaScript chạy giao diện.

Ví dụ HTML

```
<script type="application/ld+json">
{
  "@context": "https://schema.org",
  "@type": "Article",
  "headline": "Bắt đầu học HTML",
  "datePublished": "2026-11-04"
}
</script>
```

Thực hành từng bước

1. Tạo Article JSON-LD cho trang bài viết thật.
2. Đối chiếu headline/datePublished với nội dung hiển thị.
3. Kiểm tra cú pháp JSON-LD và cấu trúc bằng công cụ.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “Structured data và JSON-LD nhập môn”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Không khai báo thuộc tính không có căn cứ trên trang. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<script type="application/ld+json">
{
  "@context": "https://schema.org",
  "@type": "Article",
  "headline": "Bắt đầu học HTML",
  "datePublished": "2026-11-04"
}
</script>
```

Phân biệt dữ liệu có cấu trúc với thẻ HTML ngữ nghĩa.

Checklist

[ ] Đã đọc và giải thích được các khái niệm: Structured Data, JSON-LD, schema.org, Rich Result

[ ] Đã làm: Tạo Article JSON-LD cho trang bài viết thật.

[ ] Đã làm: Đối chiếu headline/datePublished với nội dung hiển thị.

[ ] Đã làm: Kiểm tra cú pháp JSON-LD và cấu trúc bằng công cụ.

[ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp

[ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc

[ ] Hoàn thành: Không khai báo thuộc tính không có căn cứ trên trang.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developers.google.com/search/docs/appearance/structured-data/intro-structured-data

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

SEO on-page: title, headings, link và mô tả

3. HTML và Core Web Vitals: ảnh, preload, tài nguyên

Hiểu tác động của tài nguyên và thứ tự tải.

Mục tiêu học tập

- Hiểu và giải thích được LCP trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "HTML và Core Web Vitals: ảnh, preload, tài nguyên" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

LCP · CLS · Lazy Loading · Preload · Resource Hints · Fetch Priority

Giải thích chi tiết

Vì sao cần học trong thực tế?

Hiểu tác động của tài nguyên và thứ tự tải. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- Ảnh đầu trang quan trọng thường không nên lazy load; ảnh ngoài màn hình có thể lazy load.
- Khai báo kích thước ảnh hỗ trợ ổn định bố cục, nhưng CSS vẫn quyết định hiển thị.
- preload/fetchpriority là gợi ý ưu tiên, dùng sai có thể tranh băng thông với tài nguyên quan trọng.
- script defer giúp tải script không chặn parse như script thường, nhưng thứ tự và kiểu module cần hiểu.
- Core Web Vitals đo trải nghiệm thực tế, không chỉ kích thước tệp HTML.

Ví dụ HTML

```
<img src="/images/hero.webp" alt="Học HTML thực chiến" width="1200" height="600"
fetchpriority="high">
<img src="/images/gallery.webp" alt="Buổi học" width="800" height="600" loading="lazy">
<script src="/js/site.js" defer></script>
```

Thực hành từng bước

1. Kiểm tra Lighthouse và Network trên trang có ảnh.
2. Thêm width/height rồi so sánh layout shift.
3. Thử loading lazy cho ảnh dưới màn hình, không dùng với hero.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “HTML và Core Web Vitals: ảnh, preload, tài nguyên”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Chứng minh được tối thiểu hai tối ưu bằng đo đạc, không chỉ đoán. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<img src="/images/hero.webp" alt="Học HTML thực chiến" width="1200" height="600"
fetchpriority="high">
<img src="/images/gallery.webp" alt="Buổi học" width="800" height="600" loading="lazy">
<script src="/js/site.js" defer></script>
```

Hiểu tác động của tài nguyên và thứ tự tải.

Checklist

[ ] Đã đọc và giải thích được các khái niệm: LCP, CLS, Lazy Loading, Preload, Resource Hints, Fetch Priority
[ ] Đã làm: Kiểm tra Lighthouse và Network trên trang có ảnh.
[ ] Đã làm: Thêm width/height rồi so sánh layout shift.
[ ] Đã làm: Thử loading lazy cho ảnh dưới màn hình, không dùng với hero.
[ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
[ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
[ ] Hoàn thành: Chứng minh được tối thiểu hai tối ưu bằng đo đạc, không chỉ đoán.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://web.dev/learn/performance/

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

Structured data và JSON-LD nhập môn

---

4. Giới hạn an toàn của HTML, external links và form

Không nhầm HTML client-side với bảo mật ứng dụng.

Mục tiêu học tập

- Hiểu và giải thích được HTTPS trong ngữ cảnh HTML thực tế

- Tự triển khai chủ đề “Giới hạn an toàn của HTML, external links và form” mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

HTTPS · CSP · XSS · Referrer Policy · Privacy · Input Validation

Giải thích chi tiết

Vì sao cần học trong thực tế?

Không nhầm HTML client-side với bảo mật ứng dụng. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- HTML và thuộc tính form không bảo vệ được backend nếu thiếu kiểm tra server.
- Không chèn nội dung người dùng thành HTML không được xử lý vì rủi ro XSS.
- Liên kết/iframe bên thứ ba có thể tạo rủi ro riêng tư và gửi referrer tùy chính sách.
- CSP thường được đặt qua HTTP header, một số hình thức qua meta có hạn chế; không phải mọi chính sách đều dùng meta được.
- Không thu thập dữ liệu cá nhân khi chưa có mục đích và thông tin phù hợp.

Ví dụ HTML

```
<form action="/api/contact" method="post">
  <label for="message">Lời nhắn</label>
  <textarea id="message" name="message" required></textarea>
  <button>Gửi</button>
</form>
<!-- Server phải kiểm tra và xử lý dữ liệu an toàn. -->
```

Thực hành từng bước

1. Xem tab Security/Network để kiểm tra HTTPS.
2. Kiểm tra trang có nguồn nhúng ngoài và liệt kê dữ liệu có thể chia sẻ.
3. Viết checklist trước khi triển khai form thật.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “Giới hạn an toàn của HTML, external links và form”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.
- Không đưa secrets, API key riêng hoặc token vào HTML công khai.

Tiêu chí hoàn thành

Nêu được ít nhất 5 điều HTML thuần không thể tự bảo đảm về bảo mật. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<form action="/api/contact" method="post">
  <label for="message">Lời nhắn</label>
  <textarea id="message" name="message" required></textarea>
  <button>Gửi</button>
</form>
<!-- Server phải kiểm tra và xử lý dữ liệu an toàn. -->
```

Không nhầm HTML client-side với bảo mật ứng dụng.

Checklist

[ ] Đã đọc và giải thích được các khái niệm: HTTPS, CSP, XSS, Referrer Policy, Privacy, Input Validation

[ ] Đã làm: Xem tab Security/Network để kiểm tra HTTPS.

[ ] Đã làm: Kiểm tra trang có nguồn nhúng ngoài và liệt kê dữ liệu có thể chia sẻ.

[ ] Đã làm: Viết checklist trước khi triển khai form thật.

[ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp

[ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc

[ ] Hoàn thành: Nêu được ít nhất 5 điều HTML thuần không thể tự bảo đảm về bảo mật.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Web/Security

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

HTML và Core Web Vitals: ảnh, preload, tài nguyên

---

5. HTML validation, lỗi thường gặp và DevTools

Tìm lỗi bằng công cụ thay vì sửa mò.

Mục tiêu học tập

- Hiểu và giải thích được Nu HTML Checker trong ngữ cảnh HTML thực tế

- Tự triển khai chủ đề “HTML validation, lỗi thường gặp và DevTools” mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

Nu HTML Checker · DevTools · DOM Inspector · Network · Console · Lighthouse

Giải thích chi tiết

Vì sao cần học trong thực tế?

Tìm lỗi bằng công cụ thay vì sửa mò. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- HTML parser thường tự sửa lỗi nên trang có vẻ chạy chưa chắc đúng chuẩn.
- Nu HTML Checker phát hiện lỗi cú pháp/cấu trúc; không thay kiểm tra ngữ nghĩa bằng người.
- Network cho biết 404 và response; Elements cho biết DOM hiện tại.
- Lighthouse hỗ trợ phát hiện một phần vấn đề SEO, accessibility, performance.
- Sửa từng lỗi và kiểm tra lại; tránh yêu cầu AI thay hàng loạt mà không đọc diff.

Ví dụ HTML

```
<!-- Kiểm tra bằng: https://validator.w3.org/nu/
     Không xem lỗi trên trang như bằng chứng mã HTML đúng. -->
```

Thực hành từng bước

1. Chạy trang với Nu HTML Checker.
2. Cố tình tạo id trùng, thiếu alt, lồng thẻ sai rồi đọc báo lỗi.
3. Ghi bảng bug gồm bước tái hiện, nguyên nhân, cách sửa, kết quả.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “HTML validation, lỗi thường gặp và DevTools”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Sửa hết lỗi HTML validator trong dự án thực hành. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<!-- Kiểm tra bằng: https://validator.w3.org/nu/
     Không xem lỗi trên trang như bằng chứng mã HTML đúng. -->
```

Tìm lỗi bằng công cụ thay vì sửa mò.

Checklist

- [ ] Đã đọc và giải thích được các khái niệm: Nu HTML Checker, DevTools, DOM Inspector, Network, Console, Lighthouse
- [ ] Đã làm: Chạy trang với Nu HTML Checker.
- [ ] Đã làm: Cố tình tạo id trùng, thiếu alt, lồng thẻ sai rồi đọc báo lỗi.
- [ ] Đã làm: Ghi bảng bug gồm bước tái hiện, nguyên nhân, cách sửa, kết quả.
- [ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
- [ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
- [ ] Hoàn thành: Sửa hết lỗi HTML validator trong dự án thực hành.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://validator.w3.org/

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

Giới hạn an toàn của HTML, external links và form

GIAI ĐOẠN 9

09 · Quy trình thực tế, AI hỗ trợ và dự án cuối

Ứng dụng HTML trong dự án nhóm, framework và xây sản phẩm có thể bàn giao.

1. HTML trong template, React, Next.js và SSR

Giữ nguyên tư duy ngữ nghĩa khi chuyển sang component.

Mục tiêu học tập

- Hiểu và giải thích được SSR trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "HTML trong template, React, Next.js và SSR" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

SSR · CSR · JSX · Components · DOM · Hydration

Giải thích chi tiết

Vì sao cần học trong thực tế?

Giữ nguyên tư duy ngữ nghĩa khi chuyển sang component. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- Framework vẫn tạo ra HTML/DOM cuối cùng cho trình duyệt.
- JSX giống HTML nhưng khác ở một số thuộc tính như className, htmlFor; cần hiểu cách framework ánh xạ.
- SSR tạo HTML ban đầu phía server; CSR có thể tạo DOM phía client; hydration gắn hành vi vào HTML đã render.
- Component không bảo đảm semantic HTML: div soup vẫn là div soup.
- Không cần học framework để hiểu HTML; học HTML tốt giúp dùng framework đúng.

Ví dụ HTML

```
<!-- HTML thuần: -->
<label for="email">Email</label>
<input id="email" name="email" type="email">
<!-- JSX tương ứng dùng htmlFor thay cho for. -->
```

Thực hành từng bước

1. Inspect HTML thực tế của một trang React/Next.js.
2. Viết cùng một card với HTML thuần và JSX, so sánh khác biệt.
3. Tìm lỗi button lồng a hoặc heading không hợp lý trong component.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài "HTML trong template, React, Next.js và SSR". Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Giải thích HTML gốc khác với JSX và DOM hiện tại ở đâu. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<!-- HTML thuần: -->
<label for="email">Email</label>
<input id="email" name="email" type="email">
<!-- JSX tương ứng dùng htmlFor thay cho for. -->
```

Giữ nguyên tư duy ngữ nghĩa khi chuyển sang component.

Checklist

[ ] Đã đọc và giải thích được các khái niệm: SSR, CSR, JSX, Components, DOM, Hydration

[ ] Đã làm: Inspect HTML thực tế của một trang React/Next.js.

[ ] Đã làm: Viết cùng một card với HTML thuần và JSX, so sánh khác biệt.

[ ] Đã làm: Tìm lỗi button lồng a hoặc heading không hợp lý trong component.

[ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp

[ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc

[ ] Hoàn thành: Giải thích HTML gốc khác với JSX và DOM hiện tại ở đâu.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://react.dev/learn/writing-markup-with-jsx

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

HTML validation, lỗi thường gặp và DevTools

2. Dùng AI học HTML và review code mà không lệ thuộc

Biết đặt yêu cầu, kiểm chứng và chịu trách nhiệm chất lượng HTML.

Mục tiêu học tập

- Hiểu và giải thích được AI-assisted Development trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "Dùng AI học HTML và review code mà không lệ thuộc" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

AI-assisted Development · Prompt · Code Review · Hallucination · Verification

Giải thích chi tiết

Vì sao cần học trong thực tế?

Biết đặt yêu cầu, kiểm chứng và chịu trách nhiệm chất lượng HTML. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- AI có thể phác thảo skeleton, giải thích thuộc tính, đề xuất semantic structure và gợi ý test.
- AI có thể bịa thuộc tính, dùng thẻ sai hoặc tạo form không thực sự gửi dữ liệu.
- Luôn chạy code, validate, kiểm tra bàn phím và đọc source/DOM thay vì tin AI.
- Đưa yêu cầu có ngữ cảnh, tiêu chí cụ thể, giới hạn công nghệ và đầu ra cần kiểm chứng.
- Không dán dữ liệu cá nhân thật, credentials hoặc nội dung nội bộ vào công cụ không được phép.

Ví dụ HTML

```
<!-- Prompt mẫu:
Bạn là reviewer HTML. Chỉ ra các lỗi semantic,
accessibility và validation trong đoạn HTML sau.
Với mỗi lỗi, nêu cách tự xác minh bằng MDN hoặc
Nu HTML Checker. Không thêm CSS/JS nếu không cần.
-->
```

Thực hành từng bước

1. Nhờ AI tạo trang sự kiện HTML thuần theo yêu cầu rõ ràng.
2. Tự rà 10 tiêu chí: heading, semantics, alt, label, link, path, validation, security, language, metadata.
3. Yêu cầu AI giải thích từng lỗi, đối chiếu MDN rồi sửa thủ công.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “Dùng AI học HTML và review code mà không lệ thuộc”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Tìm được tối thiểu 3 điểm cần sửa trong sản phẩm AI tạo ra. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<!-- Prompt mẫu:
Bạn là reviewer HTML. Chỉ ra các lỗi semantic,
accessibility và validation trong đoạn HTML sau.
Với mỗi lỗi, nêu cách tự xác minh bằng MDN hoặc
Nu HTML Checker. Không thêm CSS/JS nếu không cần.
-->
```

Biết đặt yêu cầu, kiểm chứng và chịu trách nhiệm chất lượng HTML.

Checklist

- [ ] Đã đọc và giải thích được các khái niệm: AI-assisted Development, Prompt, Code Review, Hallucination, Verification
- [ ] Đã làm: Nhờ AI tạo trang sự kiện HTML thuần theo yêu cầu rõ ràng.
- [ ] Đã làm: Tự rà 10 tiêu chí: heading, semantics, alt, label, link, path, validation, security, language, metadata.
- [ ] Đã làm: Yêu cầu AI giải thích từng lỗi, đối chiếu MDN rồi sửa thủ công.
- [ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
- [ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
- [ ] Hoàn thành: Tìm được tối thiểu 3 điểm cần sửa trong sản phẩm AI tạo ra.

Tài liệu tham khảo

Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Structuring_content

MDN HTML Reference Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

HTML trong template, React, Next.js và SSR

3. Capstone: Website thông tin sự kiện 5 trang HTML thuần

Dự án cuối tổng hợp toàn bộ kiến thức thành sản phẩm có thể triển khai.

Mục tiêu học tập

- Hiểu và giải thích được Capstone trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "Capstone: Website thông tin sự kiện 5 trang HTML thuần" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

Capstone · Multi-page · Form · Accessibility · SEO · Performance

Giải thích chi tiết

Vì sao cần học trong thực tế?

Dự án cuối tổng hợp toàn bộ kiến thức thành sản phẩm có thể triển khai. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- Một website tốt có nội dung thật, đường dẫn rõ, điều hướng bằng bàn phím, biểu mẫu có thông tin tiếp nhận trung thực.
- Các trang đề xuất: Trang chủ, Giới thiệu, Lịch trình, Thư viện, Đăng ký/Liên hệ.
- Bảng lịch trình phải là table có caption/th; ảnh có alt/figure, biểu mẫu có label/validation.
- SEO title/description độc nhất, tài nguyên tải hợp lý và không có broken links.
- Chỉ công bố form hoạt động sau khi kết nối endpoint thật và kiểm thử.

Ví dụ HTML

```
<!-- Cấu trúc nộp bài:
  index.html
  about.html
  schedule.html
  gallery.html
  register.html
  assets/images/
  README.md
  QA-CHECKLIST.md
-->
```

Thực hành từng bước

1. Lập sitemap + outline nội dung cho 5 trang.
2. Viết HTML và assets, không dùng framework, dùng CSS nhẹ tùy chọn sau khi HTML hoàn tất.
3. Chạy validator, kiểm thử bàn phím, DevTools và Lighthouse.
4. Viết README mô tả các quyết định semantic, báo cáo kiểm thử và công khai điểm còn hạn chế.
5. Triển khai bằng GitHub Pages hoặc hosting tĩnh, kiểm tra đường dẫn sau deploy.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “Capstone: Website thông tin sự kiện 5 trang HTML thuần”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Hoàn thành site □5 trang, không lỗi validator nghiêm trọng, điều hướng bàn phím tốt, README và bằng chứng kiểm thử. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<!-- Cấu trúc nộp bài:
  index.html
  about.html
  schedule.html
  gallery.html
  register.html
  assets/images/
  README.md
  QA-CHECKLIST.md
-->
```

Dự án cuối tổng hợp toàn bộ kiến thức thành sản phẩm có thể triển khai.

Checklist

- [ ] Đã đọc và giải thích được các khái niệm: Capstone, Multi-page, Form, Accessibility, SEO, Performance
- [ ] Đã làm: Lập sitemap + outline nội dung cho 5 trang.
- [ ] Đã làm: Viết HTML và assets, không dùng framework, dùng CSS nhẹ tùy chọn sau khi HTML hoàn tất.
- [ ] Đã làm: Chạy validator, kiểm thử bàn phím, DevTools và Lighthouse.
- [ ] Đã làm: Viết README mô tả các quyết định semantic, báo cáo kiểm thử và công khai điểm còn hạn chế.
- [ ] Đã làm: Triển khai bằng GitHub Pages hoặc hosting tĩnh, kiểm tra đường dẫn sau deploy.
- [ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
- [ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
- [ ] Hoàn thành: Hoàn thành site □5 trang, không lỗi validator nghiêm trọng, điều hướng bàn phím tốt, README và bằng chứng kiểm thử.

Tài liệu tham khảo

- Tài liệu theo chủ đề (MDN / tiêu chuẩn) Tài liệu
  https://developer.mozilla.org/en-US/docs/Learn_web_development/Getting_started/Your_first_website/Publishing_your_website
- MDN HTML Reference Tài liệu
  https://developer.mozilla.org/en-US/docs/Web/HTML
- web.dev Learn HTML Khóa học
  https://web.dev/learn/html
- Nu HTML Checker Công cụ
  https://validator.w3.org/

Nên học trước

Dùng AI học HTML và review code mà không lệ thuộc

---

4. Học tiếp CSS, JavaScript, Git và Web Standards

Biết ranh giới HTML và hướng phát triển sau khóa.

Mục tiêu học tập

- Hiểu và giải thích được CSS trong ngữ cảnh HTML thực tế
- Tự triển khai chủ đề "Học tiếp CSS, JavaScript, Git và Web Standards" mà không phụ thuộc vào code mẫu
- Kiểm tra kết quả bằng trình duyệt, DevTools, tài liệu chuẩn và tự sửa lỗi
- Biết phân biệt phần HTML tự xử lý được với phần cần CSS, JavaScript hoặc backend

Kiến thức cần học

CSS · JavaScript · Git · Web Standards · Progressive Enhancement

Giải thích chi tiết

Vì sao cần học trong thực tế?

Biết ranh giới HTML và hướng phát triển sau khóa. Đây là kỹ năng nền tảng khi xây website, kể cả lúc AI sinh phần lớn mã: người phát triển vẫn phải hiểu trình duyệt sẽ phân tích tài liệu như thế nào và người dùng có sử dụng được không.

Kiến thức cốt lõi

- CSS giải quyết bố cục/kiểu dáng, JavaScript bổ sung logic và tương tác.
- Học Git/GitHub giúp quản lý lịch sử và review thay đổi khi dùng AI.
- Progressive enhancement ưu tiên nội dung và tính năng cốt lõi hoạt động trước, rồi nâng cấp dần.
- Đọc MDN/WHATWG khi gặp tag/attribute lạ thay vì học thuộc danh sách mọi thẻ.
- Thực hành đều với sản phẩm nhỏ giúp duy trì kỹ năng tốt hơn chỉ xem video.

Ví dụ HTML

```
<!-- Bước tiếp theo đề xuất:
HTML semantic -> CSS responsive -> JavaScript -> Git/GitHub
-> Accessibility nâng cao -> Framework khi cần
-->
```

Thực hành từng bước

1. Chọn một trang trong capstone và thêm CSS responsive sau khi HTML ổn.
2. Tạo một tương tác JavaScript nhỏ như lọc lịch.

3. Đưa site lên GitHub, viết issue/PR để tự review.

Cách sử dụng AI có trách nhiệm

Có thể nhờ AI: Giải thích từng thẻ hoặc thuộc tính trong ví dụ; đề xuất tình huống thử, đưa gợi ý cải thiện cấu trúc và tạo câu hỏi kiểm tra hiểu bài.

Phải tự kiểm chứng: Đối chiếu thuộc tính với MDN/chuẩn HTML; chạy trang trong trình duyệt, kiểm tra DOM, các liên kết/tài nguyên, khả năng dùng bàn phím và các thông báo của validator.

Không nên làm: Sao chép nguyên HTML do AI tạo mà không đọc; yêu cầu AI khẳng định một biểu mẫu đã lưu dữ liệu khi chưa có backend; đưa mật khẩu, token hoặc dữ liệu cá nhân thật vào prompt.

Prompt mẫu để học với AI

> Tôi đang học bài “Học tiếp CSS, JavaScript, Git và Web Standards”. Hãy giải thích một cách ngắn gọn bằng tiếng Việt, rồi giao 3 bài tập tăng dần chỉ sử dụng HTML thuần. Đừng đưa lời giải ngay. Sau khi tôi gửi mã, hãy review theo semantic HTML, accessibility, tính hợp lệ và trích nguồn MDN để tôi tự kiểm chứng.

Sai lầm cần tránh

- Chọn phần tử chỉ vì giao diện mặc định thay vì ý nghĩa nội dung.
- Nghĩ rằng trang hiển thị được là HTML đã chuẩn, dễ tiếp cận và an toàn.
- Chỉ xem kết quả do AI tạo mà không chạy bài tập độc lập.

Tiêu chí hoàn thành

Lập được kế hoạch 3 dự án tiếp theo có tiêu chí đầu ra đo được. Có thể tự giải thích lý do lựa chọn thẻ/thuộc tính và chứng minh bằng kết quả kiểm tra.

Ví dụ minh họa

Ví dụ thực hành trọng tâm

```
<!-- Bước tiếp theo đề xuất:
HTML semantic -> CSS responsive -> JavaScript -> Git/GitHub
-> Accessibility nâng cao -> Framework khi cần
-->
```

Biết ranh giới HTML và hướng phát triển sau khóa.

Checklist

- [ ] Đã đọc và giải thích được các khái niệm: CSS, JavaScript, Git, Web Standards, Progressive Enhancement
- [ ] Đã làm: Chọn một trang trong capstone và thêm CSS responsive sau khi HTML ổn.
- [ ] Đã làm: Tạo một tương tác JavaScript nhỏ như lọc lịch.
- [ ] Đã làm: Đưa site lên GitHub, viết issue/PR để tự review.
- [ ] Đã tự kiểm tra HTML bằng trình duyệt/DevTools và validator khi phù hợp
- [ ] Đã tự kiểm chứng thông tin quan trọng từ tài liệu gốc
- [ ] Hoàn thành: Lập được kế hoạch 3 dự án tiếp theo có tiêu chí đầu ra đo được.

Tài liệu tham khảo

[Tài liệu theo chủ đề (MDN / tiêu chuẩn)](https://roadmap.sh/frontend) Tài liệu
https://roadmap.sh/frontend

[MDN HTML Reference](https://developer.mozilla.org/en-US/docs/Web/HTML) Tài liệu
https://developer.mozilla.org/en-US/docs/Web/HTML

web.dev Learn HTML Khóa học
https://web.dev/learn/html

Nu HTML Checker Công cụ
https://validator.w3.org/

Nên học trước

Capstone: Website thông tin sự kiện 5 trang HTML thuần