Phần 8

# HTML Best Practices & Advanced

Giảng viên: Kiều Tuấn Dũng

Năm: 2026

# Mục tiêu học tập

### 🌟 Best Practices 2026

Cập nhật các tiêu chuẩn và xu hướng viết HTML hiện đại nhất, đảm bảo tính bền vững cho dự án.

### 🖌️ Code Organization

Tổ chức mã nguồn sạch sẽ, dễ bảo trì và tuân thủ các quy tắc validation của W3C.

### 🚀 Advanced Tech

Làm quen với các kỹ thuật nâng cao như Web Components và tích hợp HTML5 APIs.

### ⚠️ Avoid Common Errors

Nhận diện các lỗi phổ biến thường gặp và cách khắc phục triệt để.

# HTML5 Best Practices — Code Style

## Quy tắc vàng

- **Indentation (Thụt đầu dòng):**
  Sử dụng 2 spaces mỗi cấp độ. Không dùng Tab.
- **Lowercase (Viết thường):**
  Tên thẻ và thuộc tính phải viết thường.
- **Quotes (Dấu ngoặc kép):**
  Luôn bao quanh giá trị thuộc tính bằng "".
- **Closing Tags (Đóng thẻ):**
  Luôn đóng thẻ đầy đủ để tránh lỗi nesting.

❌ **Bad Practice**

```
<SECTION>
<div CLASS=container>
<Img src=logo.png>
</SECTION>
```

✅ **Best Practice**

```
<section>
<div class="container">
<img src="logo.png" alt="Logo" />
</div>
</section>
```

# HTML validation — W3C Validator & Common Errors

## W3C Validator

Công cụ kiểm tra tiêu chuẩn chính thức

validator.w3.org

## Các lỗi phổ biến cần tránh

- **Unclosed Tags:**
  Quên đóng thẻ (ví dụ  không có ).
- **Missing Attributes:**
  Thiếu 'alt' trong <img> hoặc 'href' trong <a>.
- **Invalid Nesting:**
  Đặt block element bên trong inline element sai cách.
- **Duplicate IDs:**
  Sử dụng cùng một ID cho nhiều phần tử.
- **No DOCTYPE:**
  Thiếu khai báo <!DOCTYPE html> ở đầu file.

# HTML5 APIs — Overview

- 📍 **Geolocation:**
  Xác định vị trí địa lý của người dùng (Latitude, Longitude).
- 💾 **LocalStorage:**
  Lưu trữ dữ liệu dạng key-value ngay trên trình duyệt client.
- 🎨 **Canvas:**
  Vẽ đồ hoạ 2D/3D động bằng JavaScript.
- ⚡ **WebSocket:**
  Giao tiếp hai chiều thời gian thực (Real-time).

```
<!-- Canvas Example -->
<canvas id="myCanvas"></canvas>
const c =
document.getElementById('myCanvas');
const ctx = c.getContext('2d');
ctx.fillStyle = '#3a5bc7';
ctx.fillRect(10, 10, 150, 100);
```

*Tất cả các API này đều cần JavaScript để hoạt động. HTML5 cung cấp "móc nối" để JS tương tác với phần cứng hoặc tính năng trình duyệt.*

# Web Components — Custom Elements

Web Components cho phép tạo các thẻ HTML tuỳ chỉnh, gói gọn logic và style, có thể tái sử dụng.

## Thành phần chính:

- **Custom Elements:**
  Định nghĩa thẻ mới bằng JS class.
- **Shadow DOM:**
  Cách ly style và DOM.
- **HTML Templates:**
  Sử dụng <template> và <slot>.

```
class MyElement extends HTMLElement {
  connectedCallback() {
    this.innerHTML = `<p>Hello!</p>`;
  }
}
// Define custom element
customElements.define('my-element',
MyElement);
// HTML Usage:
<my-element></my-element>
```

💡 *Lưu ý: Đây là chủ đề nâng cao, sẽ học sâu hơn ở phần JavaScript.*

# Performance — HTML Optimization

- **Scripts Loading:**
  Sử dụng `defer` hoặc `async` để không chặn render.
- **Lazy Loading:**
  Tải ảnh/iframe khi cần thiết để tăng tốc độ load ban đầu.
- **Resource Hints:**
  Sử dụng preconnect/dns-prefetch cho tài nguyên bên ngoài.
- **Structure:**
  Giảm độ sâu DOM và sử dụng Semantic HTML.

```
// 1. Defer scripts
<script src="app.js" defer></script>
// 2. Lazy load images
<img src="pic.jpg" loading="lazy" />
// 3. Preconnect
<link rel="preconnect" href="..." />
```

# Lỗi thường gặp — Tránh các lỗi này!

- ❌ **Missing Metadata:**

  Thiếu <!DOCTYPE html>, thuộc tính lang="vi", hoặc charset utf-8.
- ❌ **Accessibility Issues:**

  Bỏ qua thuộc tính alt cho ảnh hoặc dùng thẻ không đúng ngữ nghĩa (div thay vì button).
- ❌ **Deprecated Tags:**

  Sử dụng , <font>, <marquee> (hãy dùng CSS thay thế).
- ❌ **Inline Styles:**

  Lạm dụng style="..." làm code khó bảo trì (hãy tách ra file CSS).

# Tóm tắt — Best Practices & Advanced

**Best Practices**

Clean code, format chuẩn

**Validation**

W3C Standard Compliance

**Web Components**

Custom Reusable Elements

**HTML5 APIs**

Geolocation, Canvas, Storage

**Performance**

Optimization techniques

**Error Prevention**

Avoid common pitfalls

# Tổng kết HTML5 toàn diện — 8 bài học

- **1. Introduction:**
  Cấu trúc cơ bản của trang HTML.
- **2. Text & Lists:**
  Định dạng văn bản và tạo danh sách.
- **3. Links & Media:**
  Liên kết trang và chèn ảnh/video.
- **4. Tables & Forms:**
  Hiển thị dữ liệu và thu thập thông tin.
- **5. Semantic HTML:**
  Cấu trúc trang có ý nghĩa (header, nav...).
- **6. SEO Basics:**
  Tối ưu hoá cho công cụ tìm kiếm.
- **7. Accessibility (A11y):**
  Web tiếp cận cho mọi người dùng.
- **8. Best Practices:**
  Code sạch, chuẩn và hiệu năng cao.

# Q&A

Cảm ơn các bạn đã lắng nghe!

## Liên hệ & Tài liệu

📧 dungkt@tlu.edu.vn

📖 MDN Web Docs

✅ W3C Validator