

Plano de Estudos Completo: HTML5 do Zero ao Expert

Módulos 15-17: HTML+CSS+JS, Validação & Debug, Boas Práticas & Convenções

Professor: Sandro Pereira

2025



Módulo 15
HTML+CSS+JS



Módulo 16
Validação & Debug



Módulo 17
Boas Práticas



Duração
6 aulas

MÓDULO 15 – HTML+CSS+JS (INTEGRAÇÃO)

Integrando as três tecnologias web de forma eficiente

Objetivo

Aprender a integrar HTML, CSS e JavaScript de forma eficiente, mantendo a separação de responsabilidades

Tópicos Principais

- 📦 Regras de separação: conteúdo vs apresentação vs comportamento
- { }
 Data-attributes: data-* para comunicação entre HTML e JS- 👉 Event handlers inline (evitar) vs addEventListener
- ✦ ✦ Exercício: lightbox na galeria só com HTML/CSS/JS puro

🕒 3 aulas

The gospel (according to Zeldman)

Behaviour

JS

Presentation

CSS

Structure

HTML



Manter a separação de responsabilidades facilita a manutenção, reutilização e colaboração em equipe

Separação de Responsabilidades



Três Camadas Distintas

HTML (conteúdo), CSS (apresentação), JavaScript (comportamento)

HTML: Estrutura semântica

CSS: Estilos e layout

JS: Interações e lógica



Data-Attributes

Comunicação entre HTML e JavaScript sem poluir o código

```
data-action="save" data-id="123">Guardar
```

```
element.dataset.action // "save"
```

```
element.dataset.id // "123"
```



Event Handlers

Preferir addEventListener em vez de handlers inline

```
// Evitar: onclick="doSomething()"
```

```
// Preferir:
```

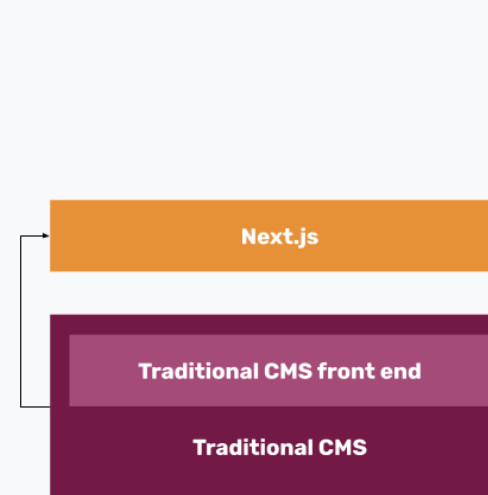
```
element.addEventListener('click',  
doSomething);
```



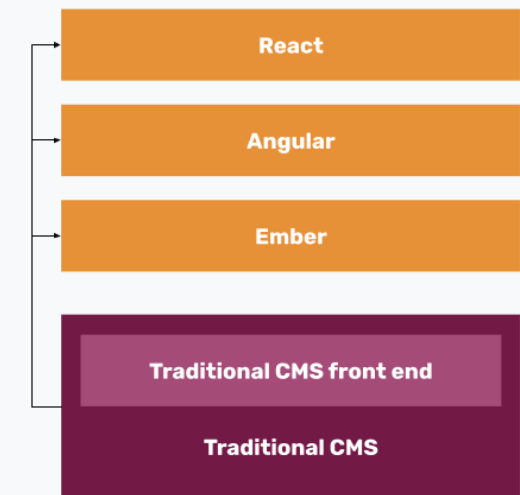
Exercício Prático

Criar lightbox na galeria usando apenas HTML, CSS e JavaScript puro

One-to-one
Before (and SMB)



One-to-many
After (and enterprise)



A separação de responsabilidades facilita a manutenção, reutilização de código e colaboração em equipe

MÓDULO 16 – VALIDAÇÃO & DEBUG

Identificando e corrigindo erros no código HTML

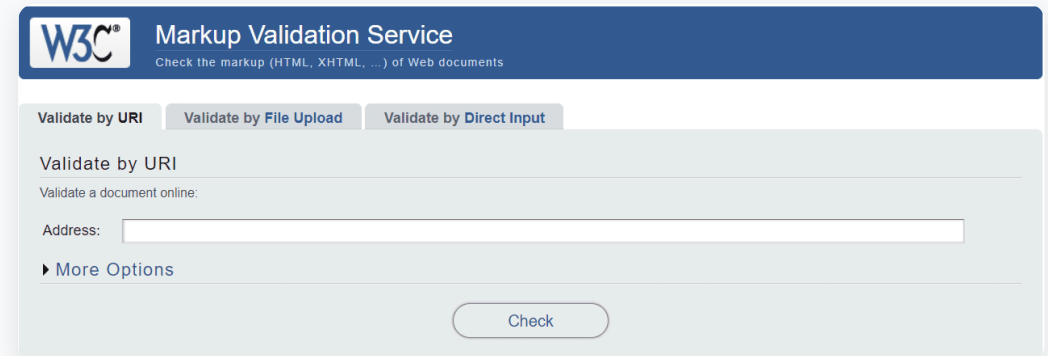
Objetivo

Aprender a validar e depurar código HTML para garantir compatibilidade e qualidade

Tópicos Principais

- ✅ W3C Validator (upload, paste, URI)
- 🔗 DevTools "Elements" → highlight
- ✂️ Linters: HTMLHint para boas práticas
- ✅ Exercício: 0 erros / 0 warnings no validator

🕒 1 aula



The screenshot shows the W3C Markup Validation Service interface. At the top, there's a blue header with the W3C logo and the text "Markup Validation Service" and "Check the markup (HTML, XHTML, ...) of Web documents". Below the header, there are three tabs: "Validate by URI", "Validate by File Upload", and "Validate by Direct Input". The "Validate by URI" tab is selected. Under this tab, there's a section titled "Validate by URI" with the instruction "Validate a document online:". Below this, there's a label "Address:" followed by a text input field. To the left of the input field, there's a link "More Options". At the bottom right of the form, there's a "Check" button.



Validar código HTML regularmente previne problemas de compatibilidade entre navegadores e melhora a acessibilidade

Ferramentas de Validação



W3C Validator

Validação oficial do padrão HTML

Métodos: upload, paste, URI

Verifica: sintaxe, acessibilidade, boas práticas



DevTools Elements

Inspecionar e destacar elementos

Atalhos: F12, Ctrl+Shift+I

Funcionalidades: highlight, inspect, console



Linters

Análise estática de código

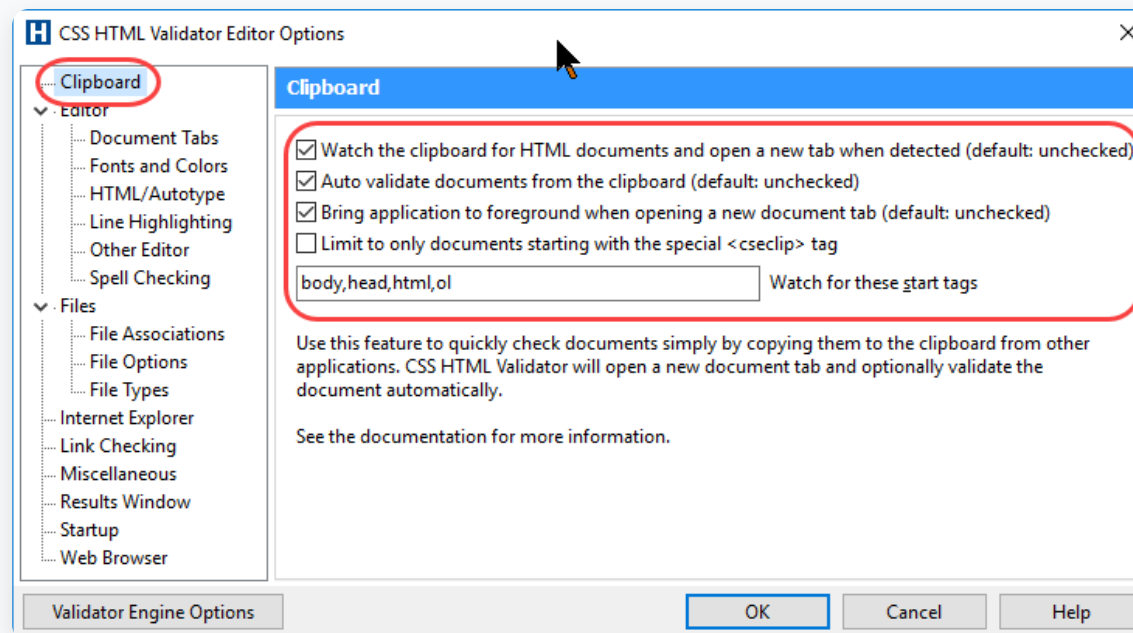
HTMLHint: verificação de boas práticas

Integração: VS Code, Sublime, WebStorm



Exercício Prático

Alcançar **0 erros / 0 warnings** no validator para todo o projeto



Validar código HTML regularmente previne problemas de compatibilidade entre navegadores e melhora a acessibilidade





MÓDULO 17 – BOAS PRÁTICAS & CONVENÇÕES

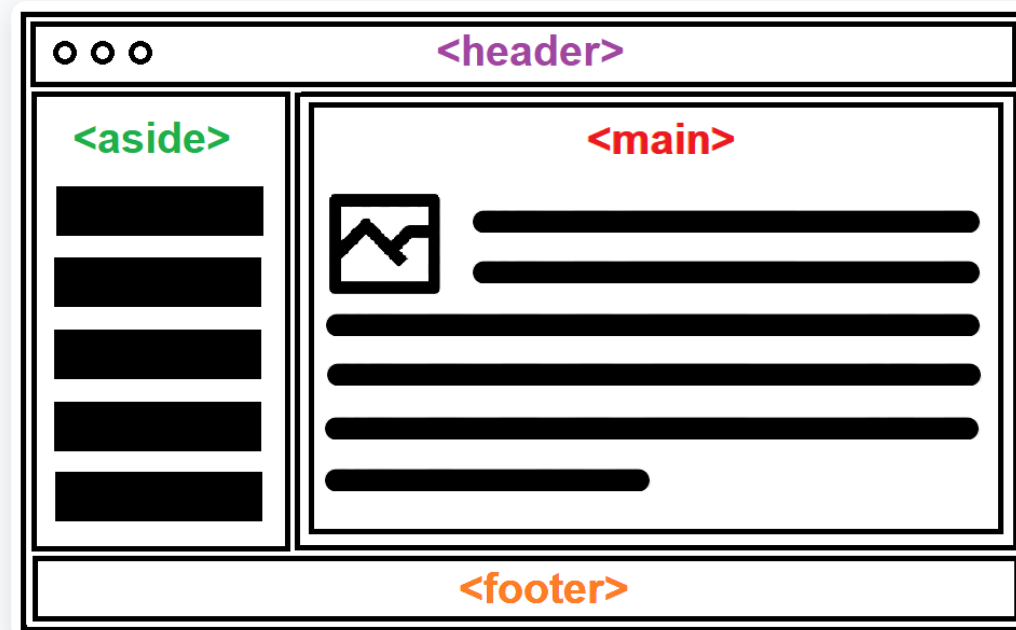
Estabelecendo padrões de qualidade e organização

Objetivo

Aprender a seguir boas práticas e convenções em HTML para código limpo e mantível

Tópicos Principais

-  Nomes de ficheiros: minúsculas e hífenes
-  Estrutura de pastas: img/, css/, js/, fonts/
-  Comentários apenas quando necessário
-  Versionamento semântico (tags git)



Código limpo e bem organizado facilita a colaboração em equipe e a manutenção futura do projeto

Convenções de Código



Nomes de Ficheiros

Use minúsculas e hífenes para consistência

Recomendado: pagina-inicial.html

Evitar: PaginaInicial.html,
pagina_inicial.html



Estrutura de Pastas

Organização lógica para facilitar manutenção

/

/img/ - imagens

/css/ - folhas de estilo

/js/ - scripts

/fonts/ - tipografia



Comentários

Use apenas quando necessário, código deve ser autoexplicativo

<header>

<!-- Navegação -->

<nav>

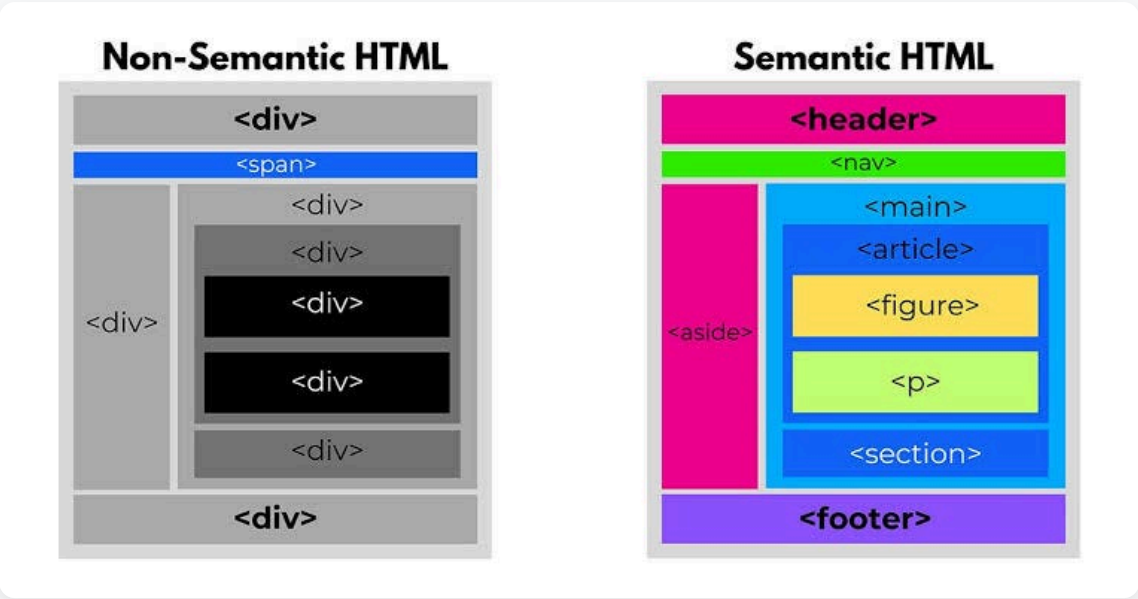
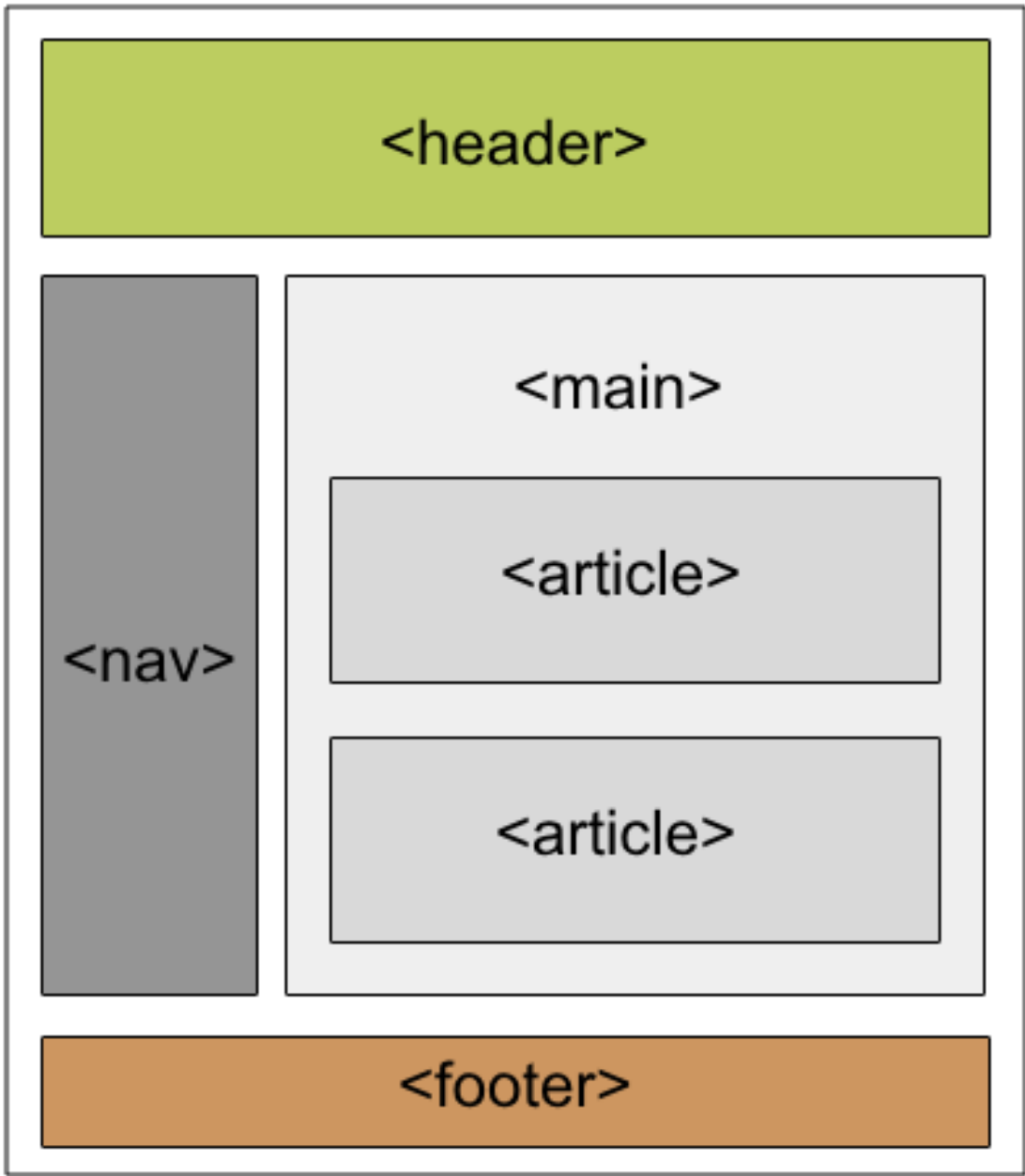


Versionamento Semântico

Use tags git para marcar versões importantes

git tag -a v1.0.0 -m "Versão inicial"

git tag -a v1.1.0 -m "Correção de bugs"



Convenções consistentes facilitam a colaboração em equipe e tornam o código mais legível e mantível

Exercício Prático

##

Objetivo

Aplicar boas práticas em um projeto existente

Passos para completar

1 Estrutura de Pastas

Revisar e reorganizar pastas seguindo convenções

2 Nomes de Ficheiros

Renomear para **minúsculas-e-hifenes**

3 Comentários

Remover desnecessários, adicionar onde o código não é autoexplicativo

4 Versionamento

Criar tags semânticas no Git: `v1.0.0`

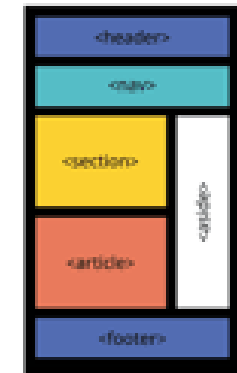
5 Validação

Verificar 0 erros no W3C Validator

STRUCTURAL MARKUP



SEMANTIC MARKUP



Código bem organizado facilita a colaboração em equipe e a manutenção futura do projeto

RESUMO DOS MÓDULOS 15-17



Módulo 15

- ✓ Separação de responsabilidades
- ✓ Data-attributes para comunicação
- ✓ Event handlers vs addEventListener
- ✓ Integração HTML+CSS+JS



Módulo 16

- ✓ W3C Validator (upload, paste, URI)
- ✓ DevTools Elements → highlight
- ✓ Linters: HTMLHint
- ✓ 0 erros / 0 warnings



Módulo 17

- ✓ Nomes de ficheiros: minúsculas e hífenes
- ✓ Estrutura de pastas organizada
- ✓ Comentários apenas quando necessário
- ✓ Versionamento semântico (tags git)

→ Próximos Módulos



Módulo 18: Ferramentas Modernas



Módulo 19: Security Headers



Módulo 20: Projeto Final