



HTML Cheat Sheet & Performance Guide

Complete Reference for Modern Web Development



1. Doctype

```
<!DOCTYPE html>
```

- Declares the document type
- Ensures the browser renders the page in **HTML5 standards mode**
- Without it, browsers may use quirks mode, causing inconsistent rendering



2. Meta Tags

Provide metadata about a webpage. Always used inside `<head>`

```
<meta charset="UTF-8"> <meta name="viewport"
content="width=device-width, initial-scale=1.0"> <meta
name="description" content="Webpage description">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta property="og:title" content="My Blog Post">
<meta name="twitter:card"
content="summary_large_image">
```



3. Tags vs Elements

Tag

Markup keyword inside angle brackets

```
<p>
```

```
</p>
```

```
<br>
```

Element

Complete structure

```
<p>Hello</p>
```



4. Attributes

Extra info applied to tags

```

```



5. Common Tags

Structure

`h1-h6``p``div``span`

Lists

`ol``ul``li``dl`

Media

`img``audio``video`

Forms

`form``input``button``select`

6. Semantic Tags

`<header>``<footer>``<main>``<section>``<article>``<aside>``<nav>``<figure>`

`<figcaption>``<time>``<mark>``<address>`

7. Forms & Inputs

```
<input type="text"> <input type="email"> <input  
type="password"> <textarea></textarea> <select>  
<option>Option 1</option> </select>  
<button>Submit</button>
```

Forms & Validation

Use `novalidate`, `required`, `pattern`,
`type="email/number/url"`



8. Accessibility (A11y)

✓ Best Practices

- Use `alt` on images
- Associate labels with inputs
- Maintain heading order
- Provide keyboard navigation
- Use roles and ARIA when needed
- Ensure color contrast $\geq 4.5:1$



9. Shadow DOM

Encapsulation for styles & DOM

```
<div id="host"></div> <script> const host =  
document.getElementById('host'); const shadow =  
host.attachShadow({ mode: 'open' }); shadow.innerHTML  
= `<p>Shadow content</p>`; </script>
```



10. ContentEditable

```
<div contenteditable="true">Edit me!</div>
```



11. Drag & Drop API

```
<div draggable="true" id="dragMe">Drag me</div>  
<script> const drag =  
document.getElementById('dragMe');  
drag.addEventListener('dragstart', e => {  
e.dataTransfer.setData('text/plain', 'dragging'); });  
</script>
```



12. Microdata & Structured Data

```
<div itemscope itemtype="http://schema.org/Person">  
<span itemprop="name">John Doe</span> <span  
itemprop="jobTitle">Frontend Developer</span> </div>
```



13. Performance Optimization

Critical Rendering Path:

1. Parse HTML → DOM
2. Parse CSS → CSSOM
3. Combine → Render Tree
4. Layout (Reflow)
5. Paint (Repaint)

Script Loading:

`<script>` : blocks parsing

`<script async>` : loads parallel, executes when ready

`<script defer>` : loads parallel, executes after parsing

Resource Loading:

```
<link rel="preload" href="styles.css" as="style">  
<link rel="prefetch" href="next-page.html"> <link  
rel="dns-prefetch" href="//example.com"> <link  
rel="preconnect" href="https://fonts.googleapis.com">
```

Images & Media

- Responsive images (srcset, sizes)
- Lazy loading: ``

- Use WebP/AVIF formats



14. Reflow & Repaint

Reflow (Layout)

Triggered when geometry/structure changes

- Size, position, font
- Margins, adding/removing DOM

Repaint

Triggered when visual style changes only

- Color, background
- Visibility, shadows

Performance Tips

- Batch DOM updates
- Use `classList` instead of multiple inline style changes
- Use `transform: translate()` for animations
- Absolute/fixed elements reflow less



15. HTML5 APIs

Canvas

localStorage

sessionStorage

Geolocation

Web Workers

WebSockets

Fetch API



16. Iframes & Embeds

```
<iframe src="page.html" sandbox loading="lazy">
</iframe> <embed src="file.pdf"> <object
data="movie.swf"></object>
```

Security: Use sandbox, loading="lazy" for security & performance



17. HTML Entities

```
< &lt; > &gt; & &amp; " &quot; &nbsp;
```



18. Progressive Enhancement vs Graceful Degradation

Progressive Enhancement

Build from semantic HTML → CSS → JS

Graceful Degradation

Build advanced app but allow fallback



19. Security

⚠ Security Best Practices

- Escape user input
- `target="_blank"` → add `rel="noopener noreferrer"`
- Use CSP (Content Security Policy)



20. Custom Data Attributes

```
<button data-id="123" data-role="admin">Click</button>
```

JavaScript: `element.dataset.id`



21. Inline vs Block Elements

Block Elements

`div``p``section`

Inline Elements

`span``a``strong`

22. Deprecated Tags (avoid)

✗ Avoid These Tags

```<center>``<marquee>````<i>`

23. Global Attributes

`id``class``style``title``lang``dir``hidden``data-*``tabindex``contenteditable`

24. Multimedia Enhancements

```
<video src="video.mp4" controls autoplay loop muted
poster="image.jpg"></video> <audio src="audio.mp3"
controls></audio> <track kind="subtitles"
src="subtitles.vtt">
```



25. Responsive & Mobile HTML

```
<meta name="viewport" content="width=device-width,
initial-scale=1.0"> <picture> <source media="(max-
width: 600px)" srcset="small.jpg">  </picture>
```



26. Intersection Observer

Purpose

Detect when an element enters/leaves viewport or a container

```
const observer = new IntersectionObserver((entries) =>
{ entries.forEach(entry => { if (entry.isIntersecting)
{ // Element is visible } }); }, { root: null, //
viewport threshold: 0.5, // 50% visible rootMargin:
'0px' });
```

Advantage: More performant than scroll events; avoids layout thrashing