UnCodemy

HTML and CSS Interview Questions PDF

HTML and CSS interviews check semantic HTML, the box model, selectors and specificity, positioning, flexbox, grid, responsive design with media queries, accessibility basics and common layout problems. This page gives short model answers with small code examples for practice, covering the questions front-end beginners meet most.

Key takeaways

- Use semantic elements such as header, nav, main and footer for structure and accessibility.
- The box model is content, padding, border and margin; box-sizing: border-box simplifies sizing.
- Specificity decides which rule wins: inline, then ID, then class, then element.
- Flexbox is for one-dimensional layout and grid for two-dimensional layout.
- Design mobile first with relative units and test with keyboard and screen reader basics.

What is HTML and CSS interview questions?

HTML structures the content of a web page with elements, and CSS controls how that content looks and is laid out, together forming the base of every website.

What is semantic HTML and why does it matter?

Semantic elements describe the meaning of content: `header`, `nav`, `main`, `section`, `article`, `aside`, `footer`, `figure` and headings `h1` to `h6`. They help screen readers, search engines and other developers understand the page. Use `div` and `span` only when no meaningful element fits. A page should have one `h1` and headings in logical order.

What is the difference between block, inline and inline-block?

Block elements start on a new line and take the available width. Inline elements flow within text and ignore width and height. Inline-block flows inline but accepts width and height.

What is the CSS box model?

Every element is a box made of content, padding, border and margin. With the default `content-box`, width applies to content only, so padding and border add to the total. With `box-sizing: border-box`, width includes padding and border, which makes layout easier to predict. Many projects apply `border-box` to all elements.

```
*, *::before, *::after { box-sizing: border-box; }
.card { width: 300px; padding: 16px; border: 1px solid #ccc; }
```

How do selectors and specificity work?

Selectors target elements by tag, class, ID, attribute, pseudo-class and combinators. When rules conflict, the more specific selector wins: inline styles, then IDs, then classes and attributes, then elements. If specificity is equal, the later rule wins. Avoid `!important` and overly deep selectors, since they make styles hard to maintain.

How does positioning work?

`static` is the default. `relative` offsets an element from its normal place. `absolute` positions it against the nearest positioned ancestor. `fixed` positions it against the viewport. `sticky` switches between relative and fixed as you scroll. Use positioning for overlays and badges, and flexbox or grid for page layout.

When do I use flexbox?

Flexbox lays out items in a row or column and distributes space. Key properties: `display: flex`, `flex-direction`, `justify-content` (main axis), `align-items` (cross axis), `gap`, and on items `flex-grow`, `flex-shrink` and `flex-basis`.

```
.nav { display: flex; justify-content: space-between; align-items: center; gap: 12px; }
```

When do I use grid?

Grid handles rows and columns together, which suits page layouts and card galleries.

```
.gallery { display: grid; grid-template-columns: repeat(auto-fit, minmax(220px, 1fr)); gap: 16px; }
```

A common rule: flexbox for components in one direction, grid for the larger two-dimensional layout.

How do I make a page responsive?

Add `<meta name="viewport" content="width=device-width, initial-scale=1">`. Use relative units (`%`, `rem`, `vw`), flexible images (`max-width: 100%`), flexible layouts and media queries for breakpoints. Start with the mobile layout and add rules for larger screens.

```
@media (min-width: 768px) { .layout { grid-template-columns: 1fr 2fr; } }
```

What are px, em and rem?

`px` is a fixed pixel size. `em` is relative to the element's font size, and `rem` is relative to the root font size. `rem` gives consistent scaling and respects user font settings.

What is the difference between display none and visibility hidden?

`display: none` removes the element from layout. `visibility: hidden` hides it but keeps its space.

What are accessibility basics?

Use semantic elements, provide `alt` text for meaningful images, associate labels with form fields, keep sufficient colour contrast, make everything reachable by keyboard with a visible focus style, and use ARIA attributes only when native elements cannot do the job. Test with the keyboard and a screen reader.

How do I centre an element?

For horizontal centring of a block with a width, use `margin: 0 auto`. To centre both ways inside a container, use `display: grid; place-items: center` or flexbox with `justify-content` and `align-items` set to `center`.

What are common layout problems and fixes?

Items overflowing a container are often fixed with `min-width: 0` on flex children or by allowing wrap. Unexpected gaps may come from margin collapsing between vertical margins. A background that does not show behind floated children can be fixed with a modern layout instead of floats. Images stretching can be controlled with `object-fit`. Explain how you debug using browser developer tools to inspect the box model and computed styles.

What forms questions are common?

Use `label` elements linked with `for`, choose the right `input` type such as email or number, add `required` and `pattern` for simple validation, group related fields with `fieldset` and `legend`, and never rely on client-side validation alone. Mention autocomplete attributes and clear error messages for accessibility.

How do I organise CSS in a larger project?

Use a naming convention such as BEM, keep components in separate files, define colours and spacing as custom properties, avoid deep nesting and keep specificity low. A consistent structure makes style changes safer.

Frequently asked questions

What is semantic HTML?

Semantic HTML uses elements like header, nav, main and footer that describe meaning, helping accessibility, search engines and maintenance.

What is the CSS box model?

Each element is a box made of content, padding, border and margin. Using box-sizing: border-box makes width include padding and border.

Flexbox or grid?

Use flexbox for layout in one direction and grid for layouts with rows and columns together.

How does CSS specificity work?

Inline styles beat IDs, IDs beat classes and attributes, and classes beat elements. With equal specificity the later rule wins.

How do I make a site responsive?

Use the viewport meta tag, relative units, flexible images and media queries, designing for mobile first.

Sources and further reading

- MDN: HTML
- MDN: CSS
- W3C WCAG overview

About this PDF

These PDFs are study material for learning and revision. They are not a certificate, a degree or a promise of a job, and technology changes, so check official documentation for the latest details. Spot an error? Write to info@uncodemy.com and the trainer team will review it.

More free resources and courses: uncodemy.com | info@uncodemy.com | +91 98183 66550