CSS Interview Questions and Answers

June 08, 2025

Introduction

This document compiles 60 CSS interview questions and answers, categorized into beginner, intermediate, and advanced levels, to aid preparation for web development interviews in 2025. Code examples are included where applicable to illustrate concepts.

1 Beginner CSS Interview Questions

1. **What is CSS?**
   CSS (Cascading Style Sheets) is a stylesheet language used to control the presentation and layout of HTML elements on a webpage.

```
body {
  background-color: #f0f0f0;
}
```

2. **What are the different ways to apply CSS to a webpage?**
   CSS can be applied via external (`<link>`), internal (`<style>`), or inline (`style` attribute) methods.

```
<link rel="stylesheet" href="styles.css">
<style>
  h1 { color: blue; }
</style>
<div style="color: red;">Inline</div>
```

3. **What is the CSS box model?**
   The box model includes content, padding, border, and margin, defining an elements size and spacing.

```
div {
  width: 200px;
  padding: 10px;
  border: 5px solid black;
  margin: 20px;
}
```

4. **What is the difference between `margin` and `padding`?**
   `margin` is the space outside the border; `padding` is the space inside the border.

```css
div {
  margin: 20px;
  padding: 10px;
}
```

5. **What are CSS selectors?**
   Selectors target HTML elements for styling, e.g., element (`p`), class (`.class`), ID (`#id`).

```css
p { color: blue; }
.highlight { background: yellow; }
#header { font-size: 24px; }
```

6. **What is the difference between `class` and `id` selectors?**
   `class` applies to multiple elements; `id` targets a unique element.

```css
.box { border: 1px solid; }
#unique { background: green; }
```

7. **What is the CSS `display` property?**
   Defines how an element is rendered, e.g., `block`, `inline`, `flex`, or `grid`.

```css
span { display: block; }
```

8. **What is the difference between `block`, `inline`, and `inline-block`?**
   `block` takes full width; `inline` flows in text; `inline-block` allows width/height settings.

```css
div { display: block; }
span { display: inline; }
button { display: inline-block; }
```

9. **What is the `position` property in CSS?**
   Controls element placement: `static`, `relative`, `absolute`, `fixed`, or `sticky`.

```css
.fixed {
  position: fixed;
  top: 10px;
  right: 10px;
}
```

10. **What is the difference between `relative` and `absolute` positioning?**
    `relative` positions relative to its normal position; `absolute` to the nearest positioned ancestor.

```css
.relative { position: relative; top: 10px; }
.absolute { position: absolute; top: 20px; left: 20px; }
```

11. **What is the CSS `float` property?**
    Places an element left or right, allowing content to wrap around it.

```
img {
  float: left;
  margin-right: 10px;
}
```

12. **How do you clear a float?**
    Use `clear` or a clearfix hack with a pseudo-element.

```
.clearfix::after {
  content: "";
  display: block;
  clear: both;
}
```

13. **What is the `z-index` property?**
    Controls the stacking order of positioned elements; higher values appear in front.

```
.overlay {
  position: absolute;
  z-index: 10;
}
```

14. **What is the difference between `relative`, `absolute`, and `fixed` positioning?**
    `relative`: to original position; `absolute`: to ancestor; `fixed`: to viewport.

```
.fixed { position: fixed; top: 0; }
```

15. **What are pseudo-classes in CSS?**
    Apply styles based on state or position, e.g., `:hover`, `:nth-child`.

```
a:hover { color: red; }
li:nth-child(2) { background: yellow; }
```

16. **What are pseudo-elements in CSS?**
    Style specific parts of an element, e.g., `::before`, `::after`.

```
p::before {
  content: " ";
}
```

17. **What is the CSS cascade?**
    Determines style application based on specificity, importance, and source order.

```
p { color: blue; }
.special { color: red; }
```

18. **What is specificity in CSS?**
    Determines rule precedence (ID > class > element).

```
#id { color: blue; }
.class { color: red; }
p { color: green; }
```

19. **What is the `!important` rule?**
    Gives a style rule the highest priority.

    ```
    p { color: blue !important; }
    ```

20. **How do you center an element horizontally and vertically in CSS?**
    Use Flexbox, Grid, or absolute positioning with transforms.

    ```
    .container {
      display: flex;
      justify-content: center;
      align-items: center;
      height: 100vh;
    }
    ```

2 Intermediate CSS Interview Questions

1. **What is Flexbox?**
   A one-dimensional layout model for aligning and distributing items in a container.

   ```
   .container {
     display: flex;
     justify-content: space-between;
   }
   ```

2. **What is CSS Grid?**
   A two-dimensional layout system for creating grid-based layouts with rows and columns.

   ```
   .grid {
     display: grid;
     grid-template-columns: 1fr 1fr;
     gap: 20px;
   }
   ```

3. **What is the difference between Flexbox and Grid?**
   Flexbox is one-dimensional (row or column); Grid is two-dimensional (rows and columns).

4. **What are media queries in CSS?**
   Apply styles based on device characteristics like screen size.

   ```
   @media (max-width: 600px) {
     body { font-size: 16px; }
   }
   ```

5. **What is the `vw`, `vh`, `rem`, and `em` units in CSS?**
   `vw`: 1% viewport width; `vh`: 1% viewport height; `rem`: root font size; `em`: parent font size.

```
h1 { font-size: 2rem; }
div { width: 50vw; }
```

6. **What is the difference between `relative` and `absolute` units?**
   Relative units (`%`, `vw`, `rem`) scale; absolute units (`px`, `cm`) are fixed.

```
div { width: 50%; }
div { width: 100px; }
```

7. **What is the `box-sizing` property?**
   Defines size calculation: `content-box` (content only) or `border-box` (includes padding and border).

```
* {
  box-sizing: border-box;
}
```

8. **How do you create a responsive image in CSS?**
   Use `max-width: 100%` and `height: auto` for scaling.

```
img {
  max-width: 100%;
  height: auto;
}
```

9. **What is the `transition` property in CSS?**
   Enables smooth changes for properties like `color` over a duration.

```
button {
  transition: background-color 0.3s ease;
}
button:hover {
  background-color: blue;
}
```

10. **What is the difference between `transition` and `animation`?**
    `transition` handles state changes; `animation` uses keyframes for sequences.

```
@keyframes slide {
  0% { transform: translateX(0); }
  100% { transform: translateX(100px); }
}
div {
  animation: slide 2s infinite;
}
```

11. **What is the `calc()` function in CSS?**
    Performs mathematical calculations, mixing units like `px` and `%`.

```
div {
  width: calc(100% - 50px);
}
```

12. **What is the `overflow` property?**
Controls excess content: `visible`, `hidden`, `scroll`, or `auto`.

```
div {
  overflow: auto;
  height: 100px;
}
```

13. **What is the purpose of the `:root` pseudo-class?**
Targets the document root for defining custom properties (variables).

```
:root {
  --primary-color: blue;
}
h1 {
  color: var(--primary-color);
}
```

14. **What are CSS custom properties (variables)?**
Store reusable values, typically defined in `:root`.

```
:root {
  --main-bg: #f0f0f0;
}
body {
  background: var(--main-bg);
}
```

15. **How do you create a sticky element in CSS?**
Use `position: sticky` with a positioning value like `top`.

```
.sticky {
  position: sticky;
  top: 0;
}
```

16. **What is the `clip-path` property?**
Clips an element to a shape or SVG path.

```
img {
  clip-path: circle(50% at 50% 50%);
}
```

17. **What is the `filter` property in CSS?**
Applies visual effects like blur or grayscale.

```
img {
  filter: grayscale(100%);
}
```

18. **What is the `transform` property?**
Applies 2D/3D transformations like rotate or scale.

```
div {
  transform: rotate(45deg);
}
```

19. **What is the difference between `visibility: hidden` and `display: none`?**
`visibility: hidden` hides but reserves space; `display: none` removes the element.

```
.hidden { visibility: hidden; }
.gone { display: none; }
```

20. **How do you create a CSS triangle?**
Use a zero-sized element with borders, setting one border color.

```
.triangle {
  width: 0;
  height: 0;
  border-left: 50px solid transparent;
  border-right: 50px solid transparent;
  border-bottom: 100px solid red;
}
```

3 Advanced CSS Interview Questions

1. **What is the CSS `object-fit` property?**
Controls how `<img>` or `<video>` content fits: `fill`, `contain`, `cover`.

```
img {
  width: 200px;
  height: 200px;
  object-fit: cover;
}
```

2. **What is the `currentColor` keyword?**
Inherits the elements `color` value for other properties.

```
div {
  color: blue;
  border: 1px solid currentColor;
}
```

3. **What are CSS logical properties?**
   Use writing-mode-aware directions (e.g., `margin-inline`) instead of physical ones.

```css
div {
  margin-inline: 20px;
}
```

4. **How do you optimize CSS for performance?**
   Minify CSS, reduce specificity, avoid universal selectors, and use critical CSS.

5. **What is the `:has()` pseudo-class?**
   Selects elements based on descendants (modern browser support).

```css
div:has(p) {
  background: yellow;
}
```

6. **What is the CSS `container` query?**
   Applies styles based on a parent containers size, not the viewport.

```css
.container {
  container-type: inline-size;
}
@container (min-width: 300px) {
  .child { font-size: 18px; }
}
```

7. **What is the `aspect-ratio` property?**
   Sets a preferred width-to-height ratio for an element.

```css
div {
  width: 200px;
  aspect-ratio: 16 / 9;
}
```

8. **How do you handle browser compatibility in CSS?**
   Use vendor prefixes, fallbacks, and tools like Autoprefixer.

```css
div {
  -webkit-transform: translateX(10px);
  transform: translateX(10px);
}
```

9. **What is the `grid-template-areas` property?**
   Defines named grid areas for layout placement.

```css
.grid {
  display: grid;
  grid-template-areas:
    "header header"
    "sidebar main";
}
.header { grid-area: header; }
```

10. **What is the `will-change` property?**
    Hints at upcoming changes to optimize rendering.

    ```css
    .animate {
      will-change: transform;
    }
    ```

11. **How do you create a CSS-only tooltip?**
    Use `::after` with `:hover` to show tooltip text.

    ```css
    .tooltip:hover::after {
      content: "Tooltip text";
      position: absolute;
      background: black;
      color: white;
      padding: 5px;
    }
    ```

12. **What is the `backdrop-filter` property?**
    Applies effects like blur to the area behind an element.

    ```css
    .overlay {
      backdrop-filter: blur(5px);
    }
    ```

13. **What are CSS subgrid features?**
    Nested grids inherit parent grid tracks for alignment.

    ```css
    .grid {
      display: grid;
      grid-template-columns: 1fr 2fr;
    }
    .subgrid {
      display: grid;
      grid-template-columns: subgrid;
    }
    ```

14. **How do you implement dark mode in CSS?**
    Use `prefers-color-scheme` or toggle classes with custom properties.

    ```css
    @media (prefers-color-scheme: dark) {
      body {
        background: #333;
        color: white;
      }
    }
    ```

15. **What is the `content-visibility` property?**
    Optimizes rendering by skipping off-screen content.

```
section {
  content-visibility: auto;
}
```

16. **What is the gap property in Flexbox and Grid?**
    Sets spacing between Flexbox or Grid items.

```
.grid {
  display: grid;
  gap: 20px;
}
```

17. **How do you create a parallax scrolling effect in CSS?**
    Use `background-attachment: fixed` or transforms.

```
.parallax {
  background-image: url('image.jpg');
  background-attachment: fixed;
  background-size: cover;
}
```

18. **What is the `scroll-behavior` property?**
    Controls scrolling animation (`smooth`, `auto`).

```
html {
  scroll-behavior: smooth;
}
```

19. **How do you handle CSS animations on low-performance devices?**
    Use GPU-friendly properties (`transform`, `opacity`) and `prefers-reduced-motion`.

```
@media (prefers-reduced-motion: reduce) {
  .animate { animation: none; }
}
```

20. **What are the benefits of using CSS-in-JS?**
    Scoped styles, dynamic styling, but may increase bundle size.

```
const Button = styled.button`
  background: blue;
  color: white;
`;
```

4 Preparation Tips

- **Master Fundamentals**: Be fluent in selectors, the box model, and positioning.
- **Learn Modern Layouts**: Practice Flexbox and Grid for complex designs.

- **Focus on Responsiveness**: Use media queries, container queries, and relative units.
- **Optimize Performance**: Write efficient CSS with minimal specificity and minification.
- **Stay Updated**: Explore new CSS features like `:has()`, subgrid, and `content-visibility`.