



CSS INTERVIEW QUESTIONS & ANSWERS

Important CSS questions and answers for
front-end developer technical interview



DEVELOPER UPDATES

- **What is CSS?**

CSS (Cascading Style Sheet) is a simple design language used to describe the look and formatting of a document written in a markup Language. It is combined with markup languages i.e. HTML & XHTML.

Using CSS we can change the color of the text, font styling, layout designs, add/ reduce spacing, add different effects for different devices and screen sizes and many more formatting.

- **What is Stylesheet?**

A stylesheet is used to build a consistent, transportable and well-designed style template. It describes the look and formatting of the document.

- **What is the latest version of CSS?**

CSS basically consists of many drafts for upcoming Standards. CSS3 is the latest version of CSS. It helps you to control the styling of the website.

- **Explain the new features in CSS3.**

CSS basically consists of many drafts for upcoming Standards. CSS3 includes some new features like New Selectors, CSS Variables and functions, Gradients, Opacity, Transitions, Transformation, Animations, FlexBox, Grid, and Media Queries.

- **Explain Different Types of Selectors**

CSS selectors are used to find the particular HTML element which you want to style or represent it in a different form.

CSS Selectors consist of different types:

1. **Universal Selector** - It is denoted using (*) in CSS. It sets the particular style for all the HTML elements within the HTML page.



```
1 * {  
2     text-align: center;  
3     padding: 12px;  
4 }
```

2. Element type Selector - The element selector selects a particular HTML tag/element from your document. This may include `<p>`, ``, `` etc.



```
1 p {  
2     font-size: 1.2em;  
3     font-weight: bold;  
4     color: rgb(210, 25, 25);  
5 }
```

3. Id Selector - The id selector is denoted with (`#`) character followed by the Id of the element. It can be used only once per page, as elements consist of a single Id value. It selects an element that has an Id set on it.



```
1 #id {  
2     font-size: 1.2em;  
3     font-weight: bold;  
4     color: rgb(210, 25, 25);  
5 }
```

4. Class Selector - The Class selector is denoted with (`.`) character followed by the class name of the element. It is used to select all the elements with a particular Class name. It sets a particular style for several HTML elements.



```
1  .class-name {  
2      font-size: 1.2em;  
3      font-weight: bold;  
4      color: rgb(210, 25, 25);  
5  }
```

5. Grouping Selector - Grouping Selector helps us to assign the same style definitions to different HTML elements. This helps in minimising our code and reduces a lot of effort rather than re-writing.



```
1  p, td, th {  
2      font-size: 1.2em;  
3      font-weight: bold;  
4      color: rgb(210, 25, 25);  
5  }
```

- **How to hide elements using CSS?**

There are 2 forms in CSS through which we can hide elements.

1. **display:none**: This removes the entire element from the Webpage.
2. **visibility:hidden**: This hides the element but the particular space is allocated to it.

- **What are the different types of CSS frameworks are used by the developers?**

1. Tailwind CSS
2. Bootstrap
3. Foundation
4. Bulma
5. UI Kit
6. Skeleton
7. Pure

- **How you can include CSS in the webpage?**

There are 3 ways to include CSS in the webpage.

1. External CSS
2. Internal CSS
3. Inline CSS

1. External CSS - External CSS is defined using `<link>` element within `<head>` section. Whenever we have to make multiple changes in pages we make use of External CSS.



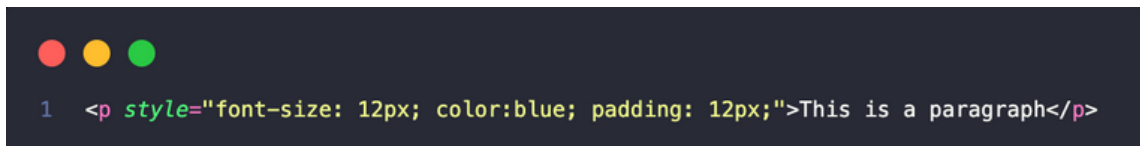
```
1 <head>
2   <link rel="stylesheet" href="style.css" />
3 </head>
```

2. Internal CSS - Internal CSS is defined using `<style>` element within `<head>` section. It helps us to design a single HTML page and change the style of the web page within HTML code.



```
1  <style>
2    .error {
3      color: red;
4    }
5    p {
6      padding: 0%;
7      font-size: 12px;
8      width: 100%;
9    }
10 </style>
```

3. Inline CSS - Inline CSS is much more different than External and Internal CSS. It directly affects the tag in which CSS is written. It applies the unique style to a particular HTML element.



```
1  <p style="font-size: 12px; color:blue; padding: 12px;">This is a paragraph</p>
```

- **What are the advantages and disadvantages of an external stylesheet?**

Advantages:

1. The style of several documents can be organised from a single file.
2. Methods like selector and grouping can be implemented to apply styles.
3. Multiple HTML elements can have several different documents, where classes can be created.

Disadvantages:

1. External documents need to be imported that carry different styles information.
2. To render your document you need to load external CSS first.

- **What are the advantages and disadvantages of an embedded stylesheet?**

Advantages:

- Classes can be created to use on various HTML tag types in the document.
- Selector and grouping methods can be used to apply styles under complex contexts
- No additional downloads necessary to receive style information

Disadvantages:

- This method can not control styles for multiple documents at once

- **Explain gradient with example**

CSS Gradient basically represents a transition between two or more colors. It is a very important aspect to beautify and add them to your content.

There are 3 types of Gradient:

1. **Linear Gradient** - In linear Gradient, the color flows from left to right, top-to-bottom, or at any angle you chose in a single direction.
2. **Radial Gradient** - Radial Gradient starts at a single point and is defined from the center position.
3. **Conic Gradient** - In Conic Gradient transition is rotated around a center point. To create a conic gradient at least 2 colors need to be defined.

- **How to override the existing CSS of the element?**

To override CSS properties of an element we make use of `!important` directive. Hence it gives the highest precedence and it overrides the cascaded property.

```
1 <h1 class="myclass">Hello World</h1>
```

```
1 h1 {  
2   color: gray !important;  
3 }  
4  
5 #myclass {  
6   color: red !important;  
7 }
```

- **Explain about vh and vw.**

1. vh(viewport height)

Represents 1% of the current height of the viewport (the visible portion of a webpage on a device's screen).

Example: If you want a div to always take up 50% of the viewport's height, no matter the device or screen size:

```
1 .div-height {  
2   height: 50vh;  
3 }
```

50% of the viewport's height

2. vw(viewport width)

Represents 1% of the current width of the viewport.

Example: If you want a div to always be 30% of the viewport's width:



```
1  .div-width {  
2    width: 30vw;  
3  }
```

30% of the viewport's width

Understanding `vh` and `vw` in CSS

The `vh` and `vw` units in CSS are related to the viewport's height and width, respectively.

I'm 50% of the viewport's height (50vh)!

I'm 30% of the viewport's width (30vw)!

Resize your browser window to see how these elements adjust their sizes based on the viewport's dimensions!

- **Explain the various positioning properties**

In CSS the position property defines how an element is positioned in the document:

1. **Position Absolute:** In this, the element will be positioned relative to its closest positioned ancestor.
2. **Position Relative:** In this, the element will be positioned relative to its original position.
3. **Position Static:** In this, the element is positioned according to the normal flow of the document. It considers the default value.
4. **Position Fixed:** In this, the element is positioned relative to the browser window. It stays in the same place even though the window is been scrolled.
5. **Position Sticky:** In this, the position is based upon the user's scroll position. It is the hybrid of relative and fixed positioning. The element is treated as relative positioned until it comes to a specified threshold, at which point it is treated as fixed positioned.

- **What are mixins?**

Mixin is a directive that lets you create CSS code that is reusable throughout the entire website. It allows defining patterns of property value pairs.

- **What is CSS Pre-processor?**

A CSS Preprocessor is a tool used to extend the basic functionality of the default Vanilla CSS through its own Scripting Language.

1. **SCSS** - Syntactically Awesome Style Sheet is the superset of CSS. SCSS is the more advanced version of CSS. Due to its advanced features, it is often termed Sassy CSS. SCSS have file extension of .scss.
2. **LESS** is a CSS preprocessor that enables customizable, manageable, and reusable style sheets for websites. LESS is a dynamic style sheet language that has the capability to extend CSS.
3. **Stylus** is a dynamic stylesheet preprocessor language compiled into Cascading Style Sheets (CSS). Its design is influenced by Sass and LESS.

- **How we can give the same style to all elements?**

To apply the same style to all the elements, we use “*” i.e. Universal Selector. It is used to set a particular style for all the HTML elements within the HTML page.



```
1  * {  
2      color:#1c76dd;  
3      text-align: center;  
4      font-size: 12px;  
5  }
```

- **Name the different types of media types.**

CSS includes different types of media types but among them the most common are:

1. **All** - This media type is used for all media type devices.
2. **Print** - This media type is basically used for printers.
3. **Screen** - This media type is used for computer screens, tablets, smartphones, etc.
4. **Screen** - This media type is used for screen readers who read the page.

- **As a front-end developer, how would you improve website performance?**

1. **Compressing and utilising lazy loading for image optimization:** We can use TinyPNG to compress images and LazyLoad to delay the loading of images until they are required. This decreases load time and reduces bandwidth usage.
2. **Reducing file sizes by minifying and bundling code. Using tools such as Webpack:** for instance, you can bundle JavaScript and CSS files and remove unnecessary characters. This reduces HTTP requests and speeds up page loading.
3. **By combining resources into a single file and utilising caching:** we can reduce HTTP requests. We can use Gzip to compress files, CSS sprites to combine them, and browser caching to store resources on the client's machine.
4. **Enable compression and implement a CDN to optimise the server:** Use tools such as NGINX to enable compression and a CDN such as Cloudflare to cache content closer to the user. This decreases latency and increases load speed.
5. **Utilize performance monitoring tools such as Google Lighthouse** to assess and identify improvement opportunities. This provides insights into best practises and optimizations for enhancing website performance and user experience.

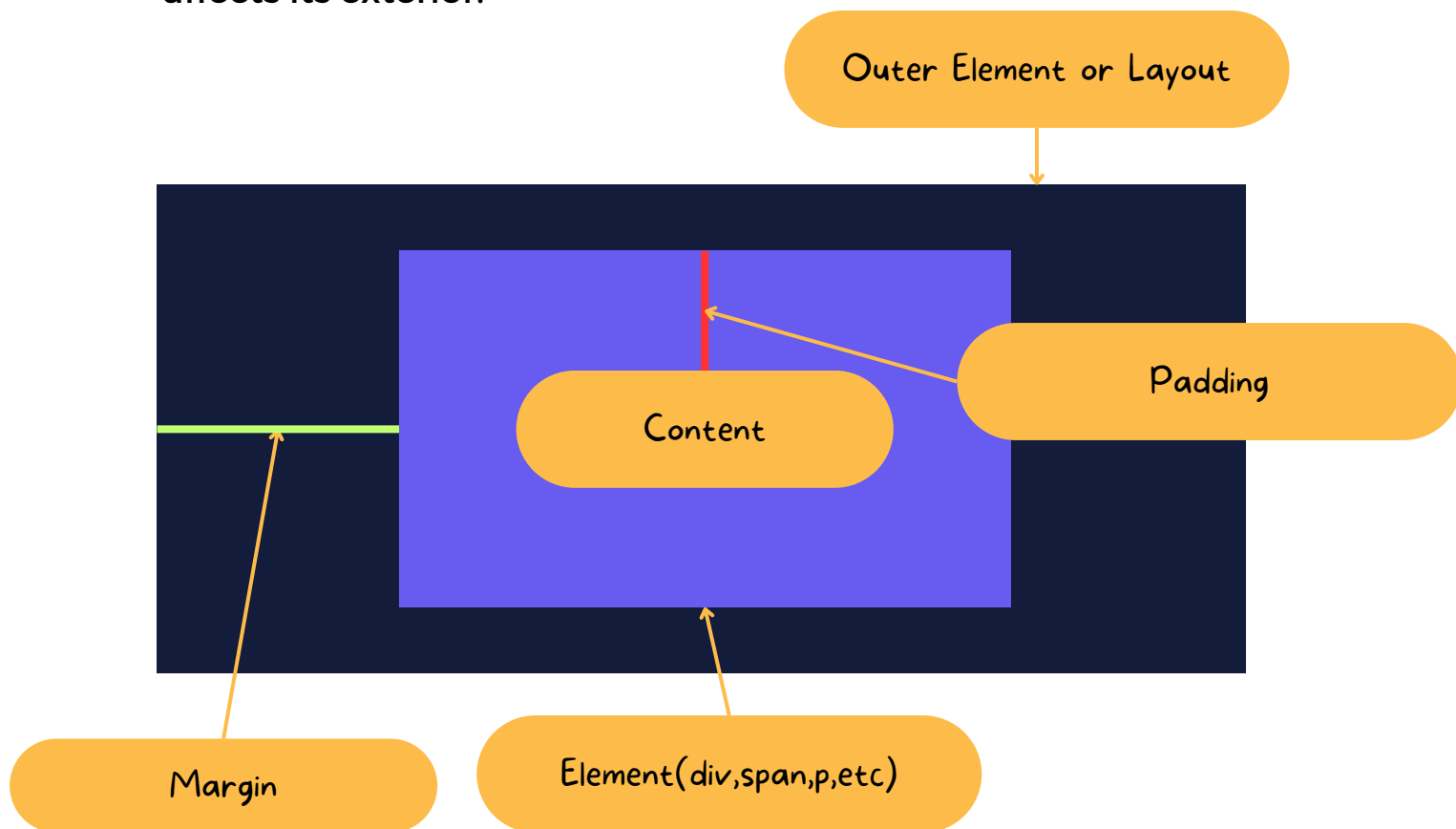
- **What is the difference between padding and margin?**

Both margin and padding are used to create space around elements in CSS, but they function differently.

Padding is the space between the content and the border of an element. It is used to expand the content area of an element.

Margin is the space between an element's border and surrounding elements. It is utilised to separate elements and control the layout of a page.

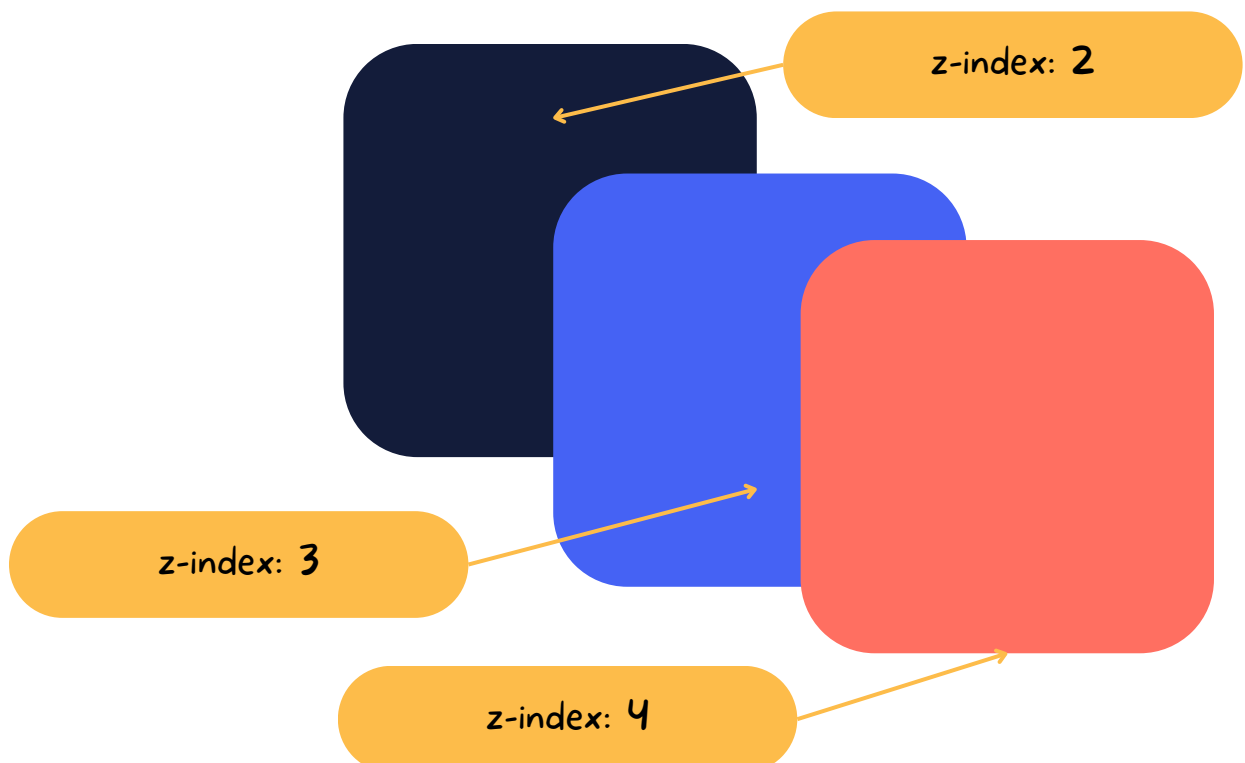
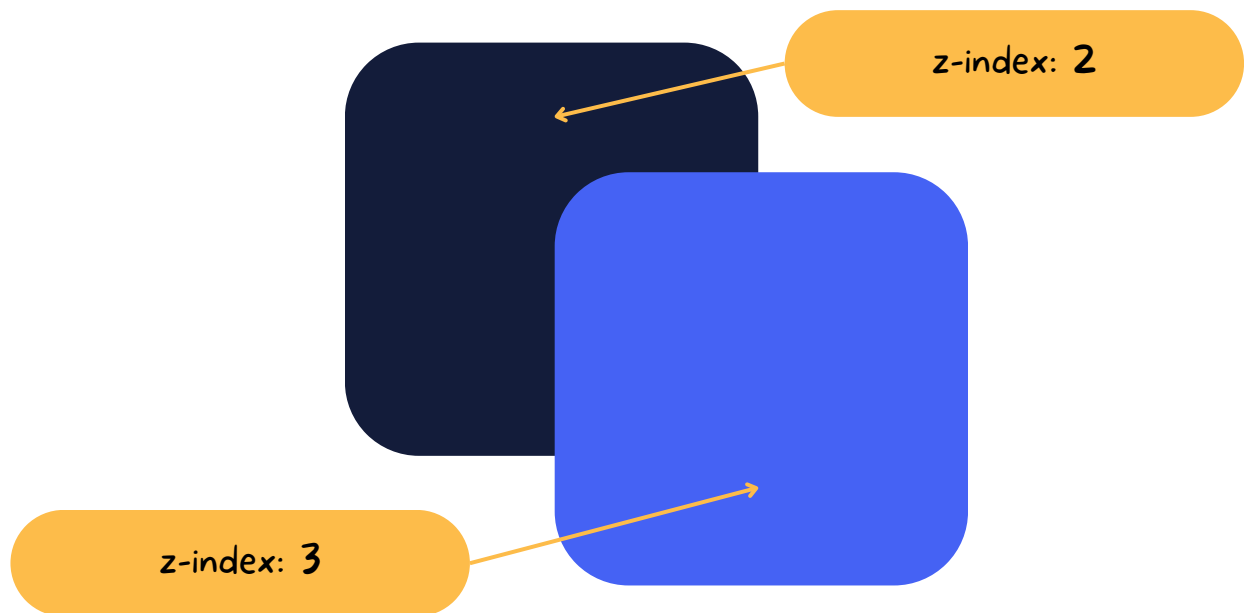
Padding affects the interior of an element, whereas margin affects its exterior.



- What is **z-index** and why it is used?

z-index is a CSS property that **controls the vertical stacking** order of elements on a web page.

It specifies the order in which elements are stacked on top of each other, with higher values appearing on top of lower values.



- **What is responsive design and how do you create it?**

Responsive web design is a way to make websites that look good and work well on many different devices and screen sizes.

This is done by designing and coding the site so that its layout, images, and other parts change automatically to fit the size of the screen being used to view it.

This method improves the user experience and makes it easier for everyone to interact with the content, no matter what device they are using.

Important steps to create responsive web design:

1. **Define the viewport:** Start by adding a meta tag to the HTML head to set the viewport, ensuring that the page is displayed correctly on different devices. Use `<meta name="viewport" content="width=device-width, initial-scale=1.0">`.
2. **Use flexbox and CSS Grid:** Use layout techniques like flexbox and CSS Grid to make more complex responsive layouts, as they give you more flexibility and control than traditional float-based layouts.
3. **Use CSS media queries:** to use different styles depending on the size of the screen, the device's resolution, or other factors. This gives you the ability to change the design for certain breakpoints.
4. **Use fluid grids:** Instead of fixed pixel widths, make layouts with relative units like percentages or viewport units (vw, vh). This will make the layout change itself depending on the size of the screen.
5. **Choose a desktop-first or mobile-first strategy:** You can start designing for smaller screens (called "mobile-first") or bigger screens (desktop-first). Mobile-first is usually recommended because it makes layouts simpler and puts the most important content first.

- **Explain about overflow property.**

With CSS's "overflow" property, you can control how content acts when it goes outside of its container.

In other words, the overflow property tells how the content inside an element should be shown when the content is bigger than the element itself.

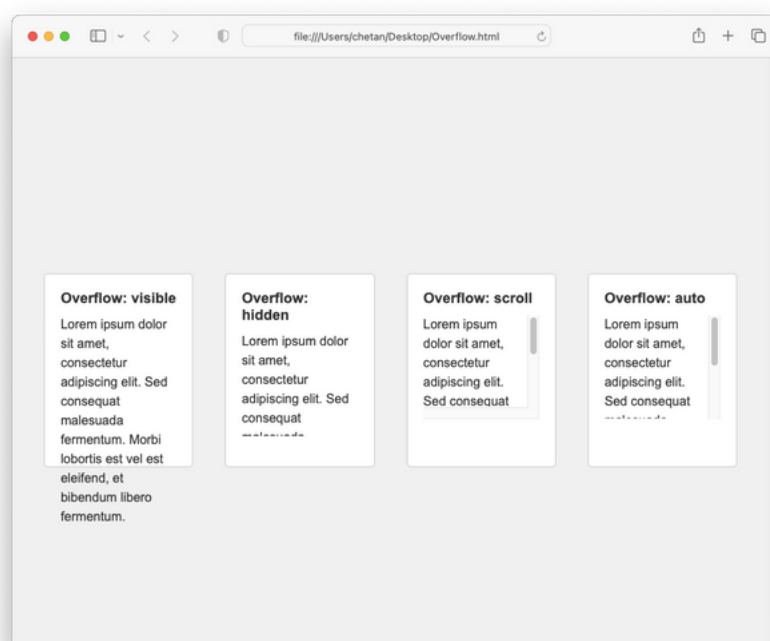
There are four common values for the overflow property:

1. **visible:** This is the default value. When content exceeds the limits of its container, it still remains visible and does not disappear.

2. **hidden:** Any content that exceeds the container's boundaries is hidden and inaccessible to the user when this value is set.

3. **scroll:** This value adds scrollbars to the container, allowing the user to scroll through the content even if it does not overflow.

4. **auto:** This value provides a more dynamic approach by adding scrollbars to the container only if the content overflows its boundaries.



- **How to animate the element using CSS?**

Animating an element using CSS involves changing the element's properties over time, such as its position, size, or opacity.

There are two main ways to do this: using keyframes and using transitions.

Keyframes allow you to define specific points in time in an animation and specify how the element should look at each point.

Each keyframe is defined as a percentage of the total animation time, from 0% (the starting point) to 100% (the ending point). You can then specify which CSS properties should be changed at each keyframe, as in the following example:

the animation changes the element's opacity and position over 2 seconds

```
1  @keyframes example-animation {
2    0% { opacity: 0; }
3    50% { opacity: 0.5; transform: translateX(50px); }
4    100% { opacity: 1; transform: translateX(100px); }
5  }
6
7  .element {
8    animation-name: example-animation;
9    animation-duration: 2s;
10   animation-timing-function: ease-in-out;
11   animation-delay: 1s;
12   animation-iteration-count: infinite;
13   animation-direction: alternate;
14 }
```

keyframe name

The animation repeats infinitely, alternating between forward and backward directions.

Transitions, on the other hand, allow you to specify which properties of an element should change, as well as the duration and timing of the animation.

They work on the principle of changing a property from one state to another state over time, as in the following example:



```
1  .element {
2      width: 100px;
3      height: 100px;
4      background-color: blue;
5      transition: all 2s ease-in-out;
6  }
7
8  .element:hover {
9      width: 200px;
10     height: 200px;
11     background-color: red;
12 }
```

- **What is the difference between `em` and `rem`?**

Both `em` and `rem` are units of measurement in CSS. They are used to define the size of the typographic elements like font-size or padding.

1.EM:

The percentage of `em` is relative to the font-size of the parent element.

For example, if the font-size of the parent element is 16px, and the font-size of the child element is set to 1.5em, then the child element font-size will be calculated as 24px. `EM` is more commonly used in typography.

2. REM:

REM stands for Root EM, meaning it's relative to the root element of the page (usually HTML).

It takes the font-size of the root element as a reference and sets the size accordingly. REM is typically used in layout and sizing.

For instance, let's say you want to create a container that will occupy 50% width of the window. If you set the width to 50rem, it will occupy 50% of the width of the viewport.

Key differences:

- EM is relative to the parent element size, while REM is relative to the root element size.
 - EM's value changes based on its parent element, and REM's value changes only once, based on the root element.
- **When to use EM or REM?**

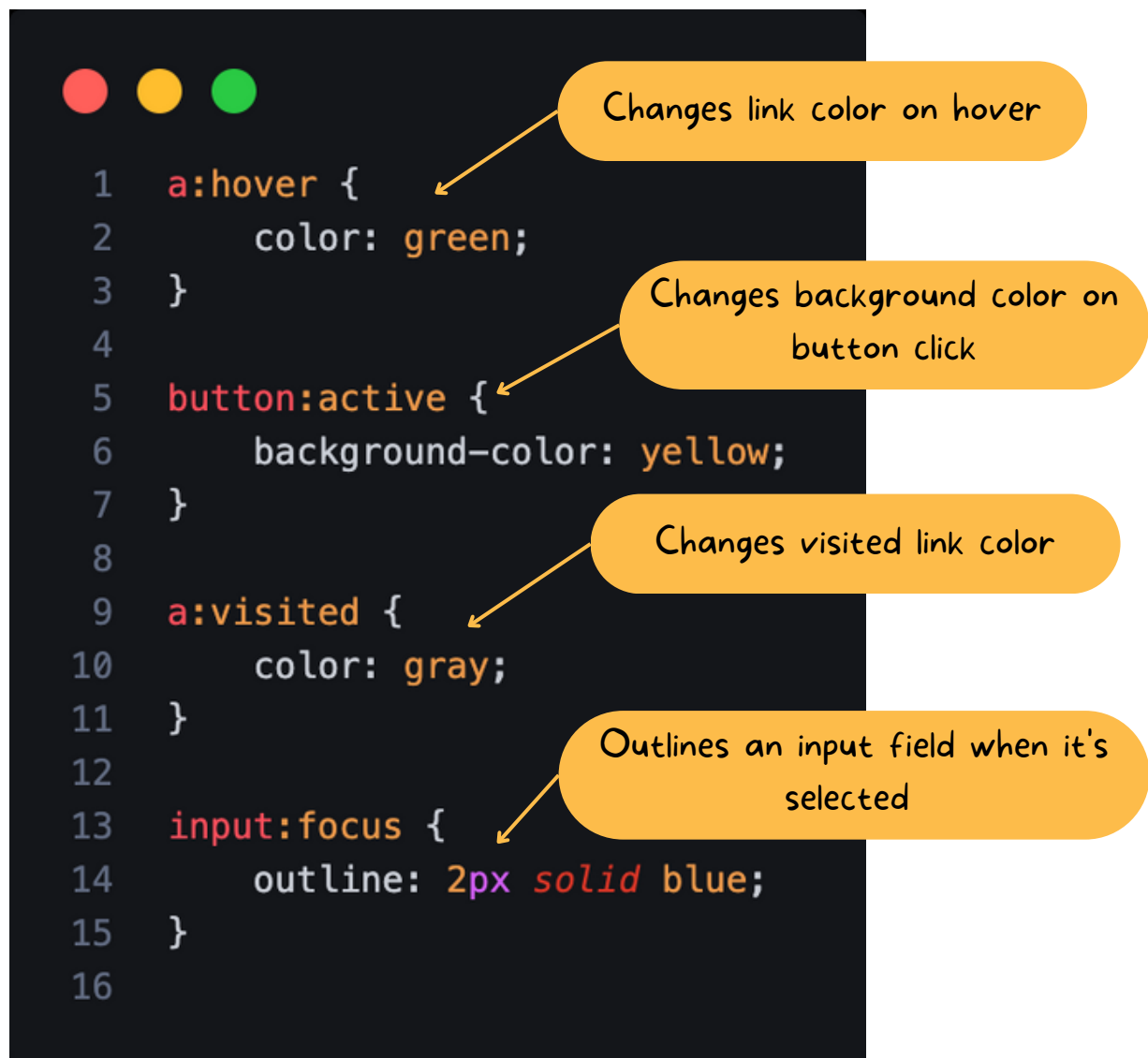
EMs are more useful for relative font size scaling. They allow for easier changes to fonts throughout a design.

On the other hand, REMs are a better option if you need to size-up structural elements like margins or padding based on the root font-size.

- **What are pseudo classes and elements?**

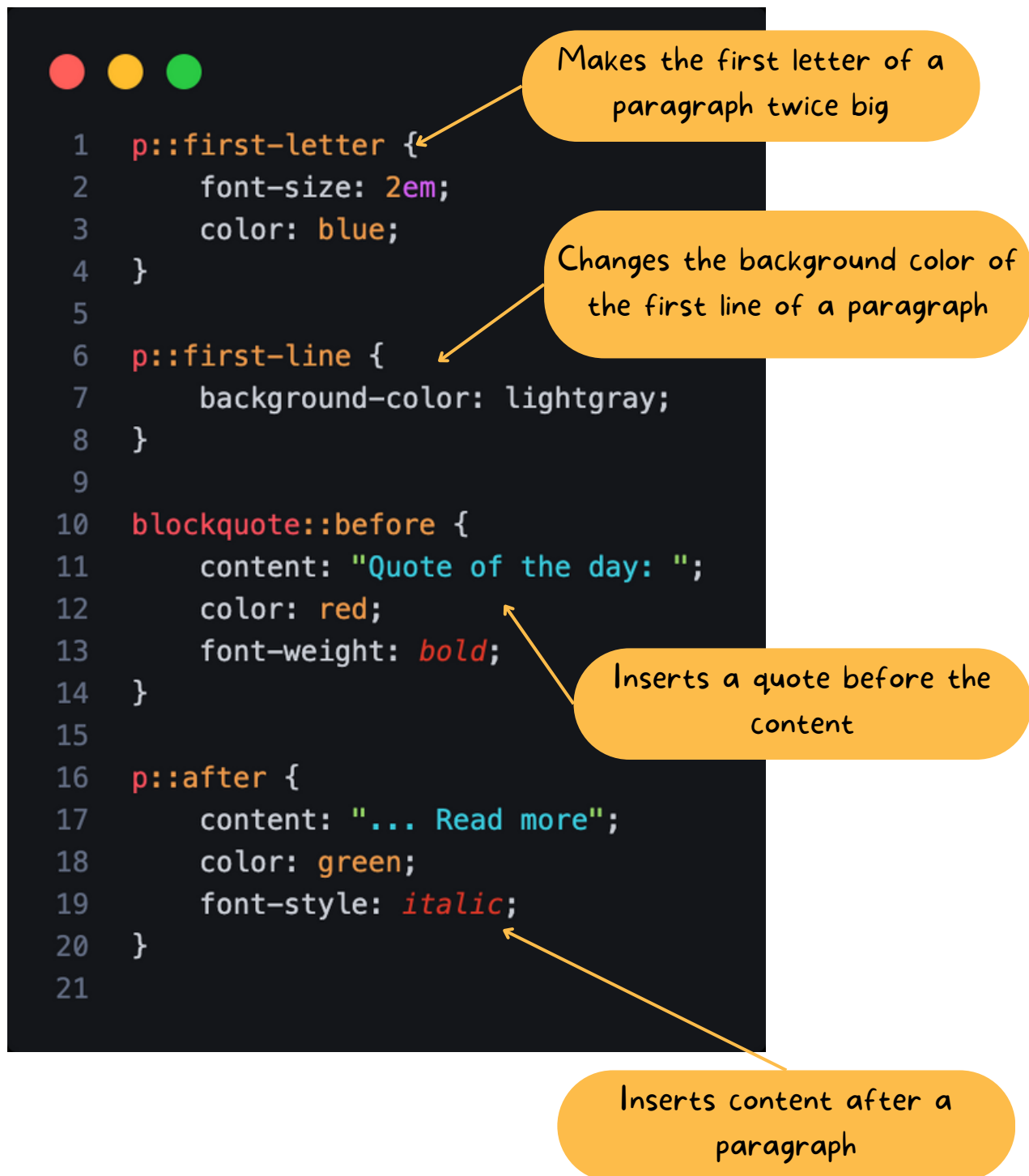
Pseudo-classes: These are special keywords added to selectors in CSS, changing the element during specific states.

1. **:hover:** Activated when you place your cursor over an element.
2. **:active:** Activated when you click an element.
3. **:visited:** Applied to links that the user has already visited.
4. **:focus:** Applied when an element is selected, like clicking an input field.



Pseudo-elements: These let you style unique parts of elements.

1. **::first-letter:** Styles the first letter of a text block.
2. **::first-line:** Styles the first line of a block of text.
3. **::before:** Inserts content before an element's content.
4. **::after:** Inserts content after an element's content.



- **What is the difference between Flexbox and CSS Grid?**

Flexbox, is a layout model that is **one-dimensional**.

This means it manages layout in a line, which can be either a row or a column, but not both at once.

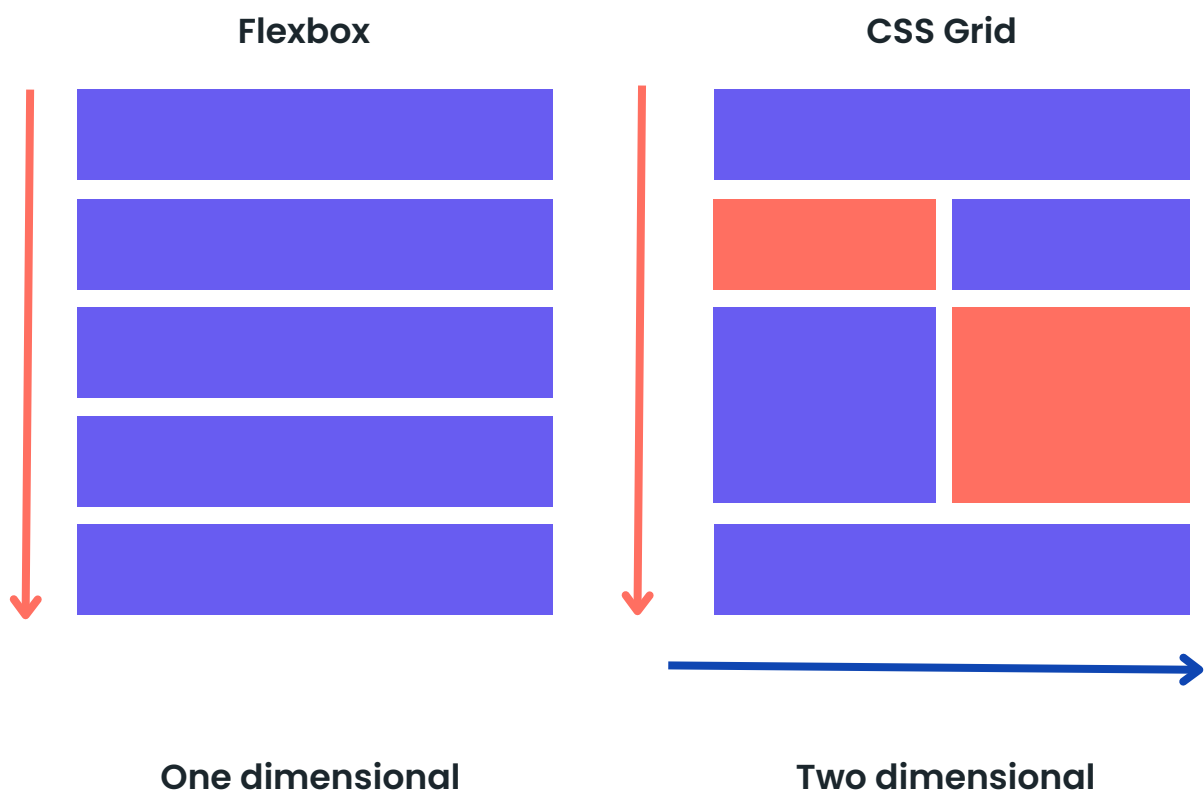
Flexbox is like having a row of pictures that you need to arrange on a table. You can decide how to spread these pictures – grouped, evenly spaced, or aligned to a side.

It's ideal for managing items in a single line.

Grid Layout is a **two-dimensional** layout system, allowing you to handle both columns and rows simultaneously.

Think of it as a chessboard, where you can arrange pieces both horizontally (rows) and vertically (columns).

This makes Grid Layout ideal for complex layouts involving multiple rows and columns.



- Tell me the difference between **inline-block**, **block**, and **inline**

1. block:

Think of it as a box that takes up the full width of its parent and starts on a new line. After the box, there's a new line.

Example - Paragraphs and headers (like `<h1>`, `<h2>`, etc.)

2. inline:

It's like a word in a sentence. It takes up only as much width as it needs and doesn't start on a new line.

Example - Text spans, links, etc.

3. inline-block:

A combination of the above two. It's like a box (like block), but it doesn't take the full width and doesn't necessarily start on a new line (like inline).

Example - Good for elements you want to sit side-by-side but also give a specific width or height.

CSS Display Properties Explained

Let's learn about the different CSS display properties: block, inline, and inline-block.

I'm a block element!

Here is an example of an inline element inside a sentence.

I'm an inline-block element!

Me too!

Notice how the block element takes the full width and starts on a new line, while the inline element sits within text and only takes up as much width as its content. The inline-block elements can have a set width and height but can sit side-by-side with other elements.

`display: block;`

`display: inline;`

`display: inline-block;`

`display: inline-block;`

- **How to set the background image to div?**

To set the background image to div element “**background-image**” CSS property is used. This property can be specified to any HTML element. We just have to provide a path to the image in **url()** value.



```
1  body {  
2      height: 400px;  
3      width: 100%;  
4      background-image: url("images/background.jpg");  
5  }
```

- **Explain the different ways to center a div horizontally and vertically in a webpage.**

Centering a div both horizontally and vertically has long been a challenge in web design.

Thankfully, with the evolution of CSS, there are now several ways to achieve this.

Here are some popular methods:

1. Using Flexbox:

Flexbox is a CSS layout module that makes it easier to design complex layout structures with a more intuitive and clean syntax.

It's now widely supported across modern browsers.

Centers child elements horizontally

```
1 .flex-container {  
2   display: flex;  
3   justify-content: center;  
4   align-items: center;  
5   height: 100vh;  
6 }
```

Centers child elements vertically

This ensures the container is full-height

The diagram shows a code editor window with three colored window control buttons (red, yellow, green) at the top left. The code is as follows:

2. Using CSS Grid:

CSS Grid is another powerful layout system, especially useful when you're building two-dimensional layouts—rows and columns.

Shorthand for align-items and justify-items

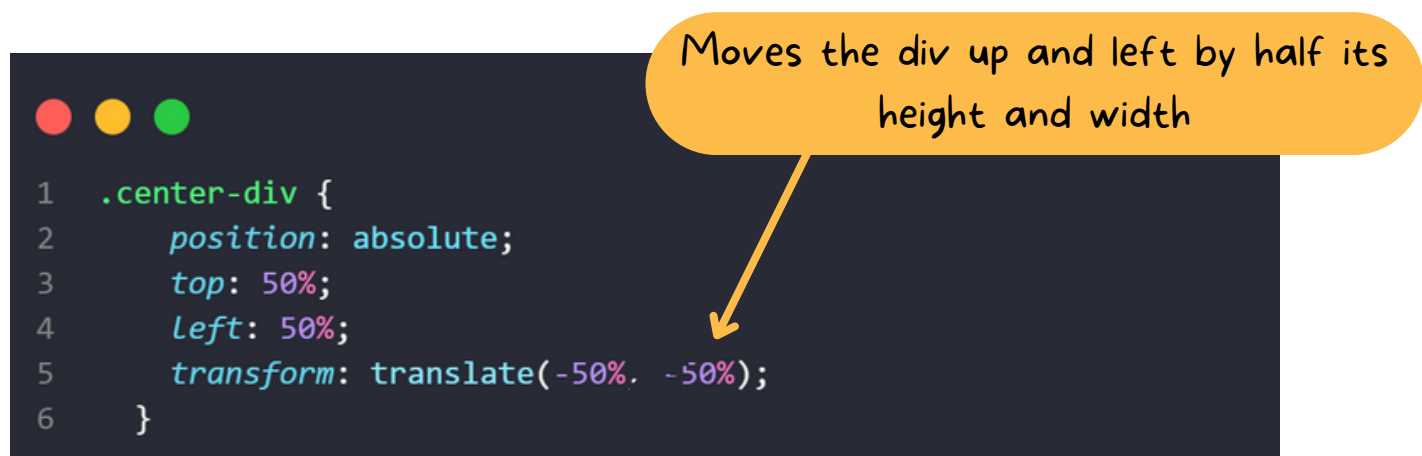
```
1 .grid-container {  
2   display: grid;  
3   place-items: center;  
4   height: 100vh;  
5 }
```

Full viewport height

The diagram shows a code editor window with three colored window control buttons (red, yellow, green) at the top left. The code is as follows:

3. Using Absolute Positioning and Transform:

This technique is great because it works irrespective of the div dimensions.



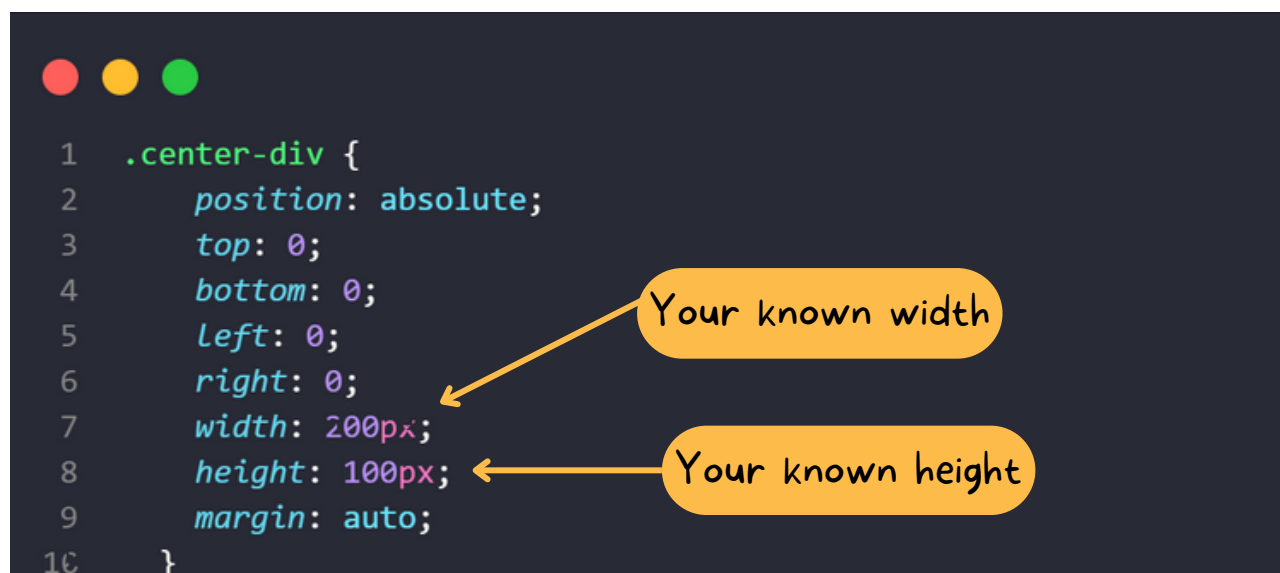
Ensure that the parent of the **.center-div** has a **position** other than **static**

(e.g., **position: relative;**) for this to work properly.

4. Using Margin Auto (with known dimensions):

If you know the width and height of the div, you can center it using margin: auto and absolute positioning.

However, this is less flexible than the previous methods because it requires knowing the dimensions beforehand.

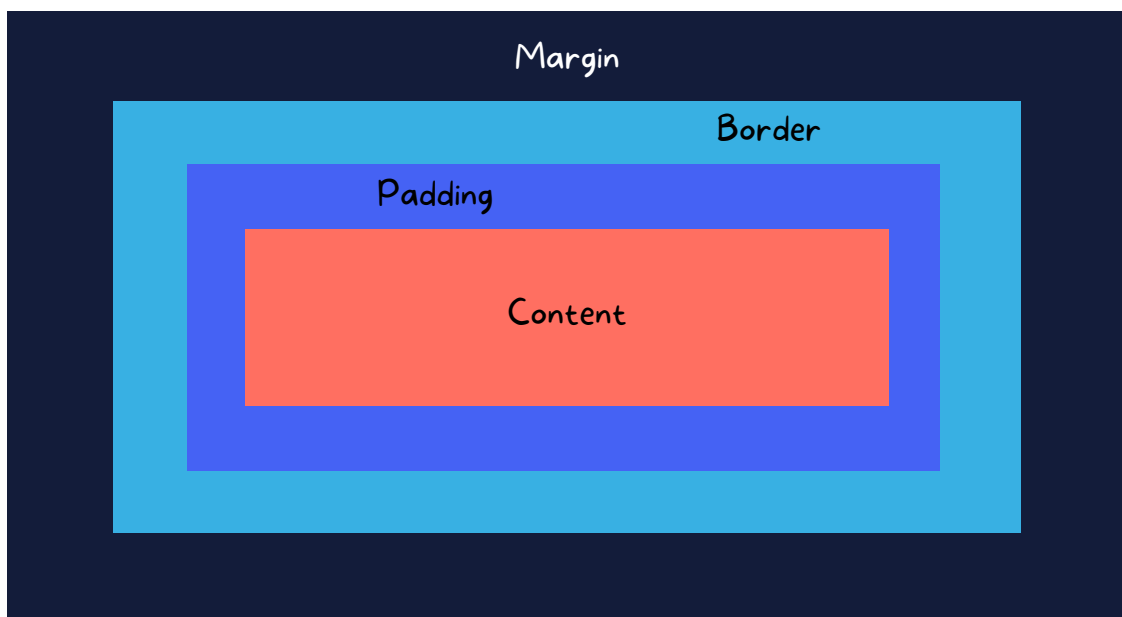


• What is the CSS box model?

Imagine every HTML element on your webpage as a box. This box has multiple layers that can be adjusted using CSS:

- **Content** - The actual content of the element, such as text, images, etc.
- **Padding** - The space between the content and the border.
- **Border** - The line that surrounds the content and padding.
- **Margin** - The space between the border and other elements next to it.

The total size of the box (width and height) is determined by adding up all these layers.



Let's say you have a **div** element, and you style it like this:

```
1  div {  
2    width: 100px;      /* Content width */  
3    height: 100px;     /* Content height */  
4    padding: 10px;     /* Padding around content */  
5    border: 5px solid black; /* 5px wide border */  
6    margin: 15px;      /* Space outside the border */  
7  }  
8
```

Here's a breakdown of the box's total width and height:

1. Content: 100px
2. Padding: 10px on the left + 10px on the right = 20px
3. Border: 5px on the left + 5px on the right = 10px

So, the total width = content + left padding + right padding + left border + right border = 100px + 10px + 10px + 5px + 5px = 130px.

Note: Margin does not add to the width/height of the box itself. Instead, it affects how far this box is from other elements.

The height would be calculated in a similar manner, also resulting in 130px in this case.

- **How do you debug CSS code?**

1. Using Browser Developer Tools:

- **Inspect Element:** Right-click on an element and select "Inspect" (or "Inspect Element"). This will open the developer tools panel where you can see the element's HTML and applied CSS.
- **Modify Styles:** Within the developer tools, you can temporarily change any CSS property to see its effect in real-time.
- **Toggle Styles:** Turn properties on or off by checking or unchecking boxes.
- **Responsive Design Mode:** View your site at different screen sizes to ensure it's responsive.

2. Check Specificity:

- Sometimes, styles aren't applied because another style with higher specificity is overriding them. Make sure to understand the basics of specificity (inline style > ID > class > element).
- If necessary, use the !important rule to override specificity, but use it sparingly.

3. By Validating CSS:

- Use online tools like the [W3C CSS Validator](#) to check for errors in your stylesheet.

4. Use a Reset or Normalize CSS:

- Browsers have default styles that can differ. Using a **reset** or **normalize.css** ensures a consistent starting point across browsers.

5. Browser Testing:

- Different browsers can render CSS differently. Test your website in various browsers (e.g., Chrome, Firefox, Safari) and versions to ensure compatibility.
- Tools like [BrowserStack](#) can help with this.

Stay ahead with daily updates!

Follow us on social media for useful web development content.



[@richwebdeveloper](#)



[@new_javascript](#)



[@developerupdates](#)



[@developerupdates](#)



[@_chetanmahajan](#)

Note: This kit is continuously updated. Please continue to check Gumroad or our website (where you purchased this) for updated questions and answers. You will receive all updates for FREE

[Download from our Website](#)

[Download on Gumroad](#)

WWW.DEVELOPERUPDATES.COM