CSS INTERVIEW QUESTIONS

1) What is the purpose of CSS media queries?

Ans:- * CSS (Cascading style sheets) media queries are a feature that allow web developers to apply different style to a document based on certain characteristics of the device or viewport.
* The main purpose of media queries is to create responsive designs that adapt to various screen sizes, resolutions & other device features.
Here are some common use cases & purpose of CSS media queries:

1) Responsive Web Design:- * Media queries are widely used in responsive web Design to make sure that web page look & function well on variety of devices like including desktops, laptops, tablets and smartphones.
* Different styles can be applied based on the screen size & width & height, ensuring a seamless user experience across different devices.

Syntax:- @media only screen & (max-width: 600px) {
/* Styles for small screens (e.g. smartphones) */
}

@media only screen and (min-width: 600px) and (max-width: 1024px) {
/* Styles for medium-sized screens (e.g. tablets) */
}

@media only screen and (min-width: 1025px) {
/* Styles for larger screens (e.g. desktops) */
}

2) Print styles:- * Media queries can be used to define specific styles for printed documents.
* This allows developers to make good for appearance of a page when it is printed by removing unnecessary elements, adjusting font sizes or changing colors.

Syntax: @media print {
/* styles for printed documents */
}

3) Orientation:- Styles can be adapted based on the position or location or orientation of the device (portrait or landscape). This is particularly useful for mobile devices that can change orientation based on how they are being held.

Syntax: @media only screen and (orientation: portrait) {
/* Styles for portrait orientation */
}

@media only screen and (orientation: landscape) {
/* styles for landscape orientation */
}

Note: Viewport means currently visible of web page browser.
* The size of viewport depends on the size of the screen
→ * The viewport is the user's visible area of a web page.

Note: portrait: The viewport is in a portrait orientation the height is greater than or equal to the width.

Note: landscape: The viewport is in a landscape orientation, the width is greater than the height.

4) Feature queries:- * Media queries can target specific features of a device such as whether it has a touch screen or supports certain CSS properties.

Syntax:- @media (hover: hover) {
/* Styles for devices with hover capability (eg desktop) */
}

@media (hover: none) {
/* Styles for devices without hover capability (eg touchscreen) */
}

By using a media queries developers can create more flexible and adaptive designs.

Q) How do you write a basic media query in CSS?

Ans:- A basic CSS media query consists of an @media rule followed by media type & one or more conditions. Here, the basic syntax.

Syntax:- @media media-type and (media feature) {
/* CSS rules to apply when the media query conditions are met */
}

* media type:- It specifies the type of media the styles apply to, such as screen for computer screens, print for printed documents, or all for all media types.
* media feature:- specifies the condition that must be true for the styles to apply. Common media features includes width, height, min-width, max-width, & more.

Ex:- here's simple example of a media query that applies styles only when the viewport width is 600 pixels or less:

/* Styles for screens with a width of 600 pixels or less */
@media only screen and (max-width: 600px) {
/* CSS rules for small screens go here */
body {
font-size: 14px;
}
}

In this ex;- * @media only screen: specifies that styles apply to the screen media type.
* and (max-width: 600px): specifies the condition the style will only apply if the maximum viewport width is 600 pixels.

You can include multiple conditions in a media query using logical operators like and or and or. Here's an ex of media query with multiple condition

Ex: /* Style for screens with width b/w 601px & 1024px

@media only screen and (min-width: 601px and max-width
/* CSS Rules for medium-sized screens go here */
body {
font-size: 10px;
}
}

3) Explain the difference b/w max-width and min-width in media queries?

Ans: * In CSS media queries both max-width and min-width are used to set conditions based on the width of the viewport or device screen. However they are used in slightly different ways.

1) Max-width:-
* It sets the max-width the type of styles inside the max-width the media query will apply.
* If the viewport width is less than or equal to the specified max-width value the styles will be applied.
* It's commonly used for making the styles that are specific to smaller screens.

Syntax Ex:- /* Styles apply if the viewport width is 600 pixels or less */
@media only screen and (max-width: 600px) {
/* CSS Rules for small screens go here */
}

2) min-width:- * It sets the minimum-width for which styles inside the media query will apply.
* If the viewport width is greater than or equal to the specified min-width value, the styles will be applied.
* It's commonly used for making a styles that are specific to larger screens or landscape.

4) What is the purpose of the viewport meta tag in responsive web design?

Ans: * The viewport meta tag is important element in responsive web design as it helps control how a webpage is displayed on various devices with different screen sizes & resolutions.
* It provides instruction to the browser on how to scale & render the content to ensure a consistent & user-friendly across different device.
* The viewport meta tag is typically placed within the <head> section of HTML document
Ex: <meta name = "viewport" content = "width = device-width, initial-scale = 1.0">

5) How can you apply different styles based on the orientation of the device - whether it's in

5) How can you apply different styles for landscape and portrait orientations using media queries?

Ans. * You can use CSS media queries to apply different styles based on the orientation of the device - whether it is landscape or portrait mode.
* The orientation of media feature is used for this purpose
Here is example

```
/* styles for portrait orientation */
@media only screen and (orientation: portrait) {
    /* CSS rules for portrait orientation go here */
    body {
        background-color: lightblue;
    }
}
/* style for landscape orientation */
@media only screen and (orientation: landscape) {
    /* CSS rules for landscape orientation go here */
    body {
        background-color: lightgreen;
    }
}
```

6) Explain the concept of a mobile-first approach in responsive design?

Ans: * The mobile-first approach is a design strategy in responsive web design that prioritizes designing & developing a website or web application for mobile devices first.
* And then progressively enhancing & adding more features for larger screens such as tablets & desktops.

** Ex: /* Styles for small screens (mobile-first) */
@media only screen and (max-width: 600px) {
body {
font-size: 10px;
}
}
/* Media query for larger screens (tablet & desktop) */
@media only screen & (min-width: 601px) {
body {
font-size: 18px;
}
}

7) What are common breakpoints used in responsive design?

Ans: * Breakpoint in responsive design are specific point at which layout or styling of web page is adjusted to provide an optimal user experience for different screen sizes.
* The choice of breakpoints depends on the content and design of the website.
Here are some common breakpoints used in responsive design:
1) Small screens (mobile phones):
* Breakpoint: Typically around 320px to 480px
* Common media query: @media only screen & (max-width: 480px) {
}
2) Medium screens (Tablets):
* Breakpoint: Typically around 601px to 768px
* Common media query: @media only screen & (min-width: 601px) & (max-width: 768px) {
}

3) Larger screens (Desktops & laptops):
* Breakpoint: 769px & above.

4) Extra large screens:
* Breakpoint: Custom depending on the design & content (1200px)

5) Orientation changes (portrait & landscape):
* Breakpoint: using the orientation media feature.

8) What is the purpose of the rem unit in media queries?
Ans * The rem unit in CSS stands for "root em" & it is a relative unit of measurement
* The value of 1 rem is equal to the font size of the root element of the document,
* Which is typically the <html> element.
* The rem unit is useful in media queries because it allows for a more flexible & scalable approach when defining styles based on the root font sizes.
* Here's an ex of how you might use rem in a media query.

Ex: /* Default font size for the root element */
html {
font-size: 16px;
}
/* media query using rem units */
@media only screen & (min-width: 40rem) {
/* style for screens with a width of 40 times the root font size or more */
body {
font-size: 1.2rem;
/* other styles for larger screens go here */
}
}

9) How can you combine multible media queries in CSS?

Ans * In CSS you can combine multible media queries by simply chaining them together with commas.
* When you use commas between media queries the styles in the curly braces will apply if any of the specified conditions are met.
* This allows you to create a set of styles that apply to multible scenarios.
* Separating with them with commas.

10) What is significance of the all keyword in media queries

Ans:- * In media queries the all keyword is used to specify that the styles within the media query should apply to all devices & media types.
* The all keyword is often included by default when writing media queries, but it can be omitted without affecting the functionality.
Here is ex:-

```
/* Media query using the "all" keyword */
@media all and (max-width: 600px) {
  /* Styles for screens with a width of 600 pixels or less */
  body {
    font-size: 14px;
  }
}
```

* In this example all is used to target all devices, regardless of their media type (e.g., screen, print) or other characteristics.
* The 'and' keyword is used to combine the media type with the specified condition (max-width: 600px).
* The 'all' keyword is the default value for media types in media queries.

1) How do you use media queries to apply styles only for print stylesheets?

Ans * To apply styles specifically for print stylesheet you can use a media query that target the print media type.
* That make sure that the style is applied in media query only take effect when the document is being printed

Ex:

```
/* styles for print stylesheets */
@media print {
    body {
    color: #333; /* Example color for text when printed */
    }
    /* Additional styles go here */
}
```

* @media print → The style should be applied only when the document is being printed.

2) What is the difference b/w screen and print in media queries?

Ans * In CSS media queries the screen and print media types are used to target different output devices.

1) Screen:-

* The screen media type is the default for stylesheets and is intended for devices with a screen or similar visual display such as computer monitors, smartphones, tablets & TVs
* Styles within a screen media query will apply when the content is being presented on a screen.
* Ex:

```
/* styles for screens (default media type) */
@media screen {
    /* Screen-specific styles go here */
    body {
        font-size: 16px;
    }
}
```

d) Print:

* The print media type is used for styles that should be applied when a document is being printed. i.e. this type is a physical printer or a browser's print functionality.

Ex:-
```
/* Styles for print stylesheets */
@media print {
    /* Print-specific styles go here */
    body {
        color: black;
    }
}
```

13) How can you hide an element on a specific screen size using media queries?

Ans:- * By adjusting its display property. To hide an element typically set the display property to none. Here an example of how can hide an element for screens with a maximum width of 600 pixels.

```
/* Hide the element on screens with a width of 600px or less */
@media only screen and (max-width: 600px) {
    .element-to-hide {
        display: none;
    }
}
```

14) Explain the role of the orientation property in media queries?

Ans:- used to target specific styles based on the orientation of the device, whether it is portrait or landscape mode. This is particularly relevant for mobile devices that can be held in different orientations. such as smartphones & tablets.

The orientation property takes on of two values

* Portrait: Applies styles when the height of the viewport is greater than or equal to width. This is typically the orientation when the device is held vertically.

/* Styles for portrait orientation */
@media only screen and (orientation: portrait) {
/* CSS rules for portrait orientation go here */
}

landscape:- Applies styles when the width of the viewport is greater than or equal to the height. This is typically the orientation when the device is held horizontally.

/* Styles for landscape orientation */
@media only screen and (orientation: landscape) {
/* CSS rules for landscape orientation go here */
}

15) How do you target specific devices using media queries?

Ans Targeting specific devices using media queries involves using a combination of media features and values to match the characteristics of the device you want to style. Media queries can be based on various criteria such as screen width, height, resolution, & orientation.

Ex 1: Targeting a specific screen width Range.

@media only screen & (min-width: 320px) & (max-width: 480px) {

}

Ex: 2 Targeting a specific device orientation

@media only screen and (orientation: portrait) {

}

16) What is the purpose of the not keyword in media queri

Ans:- The not keyword in CSS media queries is used to negat the specified conditions allowing you to target devices that do not meet a particular criteria. It provides a way to exclude certain devices or condition.

Syntax:

```
@media not screen and (max-width: 600px) {
}
```

17) How can you use media queries to adjust font sizes for different screen sizes?

Ans. You can use media queries to adjust font sizes bas on different screen sizes, ensuring a responsive desig that looks good on a variety of devices. media quer allow you to apply specific styles when certain condi such as screen width are met

Ex:- @media only screen & (max-width: 600px) {

```
body {
    font-size: 14px
}
}
```

18) What is the box sizing property in CSS and what does control?

Ans. The box-sizing property in CSS is used to control how the total width and height of an element are calculated. It determines whether the specified width and height prope include the element's padding & border or not

* The box-sizing property can take one of the following va

1) content-box (default):
* This is the default value.
* It calculates the total width & height of an element excluding padding & borders
* Ex:- box sizing: content box;

2) border-box:
* It includes the padding & border in the total width and height of an element
ex:- box-sizing: border-box.

19) Explain the difference b/w box-sizing: content box; & box-sizing border-box;

Ans: * The box sizing property in CSS control how the total width & height of an element are calculated, specifically whether or not padding & border are included in those dimensions.
* The two commonly used values for box-sizing are content box & border-box..

1) box-sizing: content-box:
* This is the default value if not explicitly set
* The width and height properties set the dimensions of the element's content area only, excluding padding & borders
* ~~The total~~
ex:- box-sizing: content box;

box-sizing: border-box;
* In this model the width & height properties set the dimensions of the element, including padding & borders
code ex:- box sizing: border box

21) Why might you choose box-sizing: border-box; as the default box model for your project?

Ans:- Choosing box-sizing: border-box; as the default box model for your project is often recommended for several reasons,

1) Easier sizing calculations:
* With box-sizing: border-box; the width & height properties directly control the dimensions of the entire element, including padding & borders.

2) Predictable layouts:
* Elements using border-box maintain a consistent size regardless of changes to padding or border values.

3) Responsive Design:
* In responsive design where layouts need to adapt to different screen sizes and orientations, using border-box simplifies the process.

4) Global consistency: Setting box-sizing: border-box; as the default can provide global consistency across the project.

5) Less CSS code for sizing:
* Developers can write less CSS code when sizing elements padding & border values don't need to be subtracted from the specific width or height, leading to cleaner & more concise stylesheets

22) Difference b/w normalizing and resetting?
Ans:- * Normalizing and resetting are two different approaches address inconsistencies and differences in default styles across different browsers.

Resetting: 1) objective:- * Complete Reset: Resets all default styles for all HTML elements to a consistent baseline
* Zeroing out styles: Typically involves setting margins, padding, & other properties to zero to remove any browser specific styling.

2) Implementation:
* Specificity: Reset styles are usually applied using a global CSS rule or a style sheet that targets all elements often through a CSS reset library.
* Zeroing styles: It often involves setting styles to zero removing default margin, padding & borders.
3) Ex:-

```
*{
margin:0;
padding:0;
box-sizing: border-box;
```

Normalizing:-
1) Objective:- * Consistent Base styles: Aim to make default styles more consistent across different browsers rather than removing them entirely.
* Preservation of useful styles: Attempts to keep certain default styles that are deemed helpful & consistent
2) Implementation:-
* Selective normalizations:- Styles are applied selectively to specific elements to normalize their appearance.

What is combinator and how is it used in a selector?
A combinator is a character that specifies the relationship between selectors, indicating which elements should be selected based on their relationship in HTML document tree.
Descendant combinator (space);
Child combinator (>);
Adjacent sibling combinator (+);
General sibling combinator (~)

24) Differentiate b/w descendant & child combinators in CSS selectors Provide examples for each?

Ans Descendant combinator (space) & child combinator (>) are both CSS combinators used to define relationships b/w selectors However they differ in how they select elements based on their position in the HTML document tree

Descendant combinator (space):

* Syntax: AB selects all element B that are descendants of an element A no matter how deeply nested.

* Ex:- div p {
    color: blue;
}

Child combinator (>):

* Syntax: A > B selects all elements B that are direct children of an element A.

Ex:-

div > p {
    color: red;
}

25) Explain the purpose of the adjacent sibling combinator (+) in CSS. Provide a use case.

Ans. The adjacent sibling combinator (+) in CSS is used to select an element that is immediately preceded by a specified element it targets an element that shares the same parent & comes directly after the specified element in the HTML document

Syntax:

A + B

* Selects element B that is an adjacent sibling of element A.

ex: h2 + p {
    color: red;
}

Purpose:- The adjacent sibling combinator is used when you want to apply styles specifically to an element that directly follows another specific element based on their relative position in the document structure.

2b) How does the general combinator (~) differ from the adjacent sibling combinator (+)?

Ans:- The general sibling combinator (~) & the adjacent sibling combinator (+) are both CSS combinator used to select elements based on their relationship to a specified element.

Adjacent sibling combinator (+):

* Syntax: A+B selects an element B that is immediately preceded by an element A

* Ex:

```
h2+p{
    color: red;
}
```

* Behavior:

* It targets only the element that is immediately preceded by an <h2>.

General sibling combinator (~):

* Syntax: A~B selects all elements B that are siblings of element A & share the same parent regardless of their position

Ex:

```
h2~p{
  color: blue;
}
```

Q7) What is the significance of the child combinator (>) in CSS selectors?

Ans:- The child combinator (>) in CSS selectors is used to select all direct children of a specified element. It establishes a parent-child relationship, ensuring that only elements that are immediate children of the specified

parent are selected excluding elements that are nested further down the hierarchy.

Syntax:

A > B

* Select all element B that are direct children of an element A.

Ex:-

```
ul > li {
}
```

Significance:

1) Direct child Relationship:
* The child combinator specifically target elements that are immediate children of the specified parent element.

2) Preventing Descendent selection:
* It excludes elements that are not direct children of the specified parent preventing styles from being applied to nested elements deeper in the hierarchy.

3) Creating specific styles for specific structure
* It allows for more specific targeting & styling

2) Provide an ex of using the descendant combinator to style nested elements

Ans: Certainly. The descendant combinator in CSS is used to select all elements that are descendants of specified element regardless of how deeply nested they are in the HTML structure.

```
div p {
}
```

Ex:

```
<div>
    <p> paragraph 1 </p>
    <p>  </p>
</div>
```

30) Explain the concept of CSS pseudo-selector Provide examples of commonly used pseudo-selector & their purpose.

A) CSS pseudo-selector are used to select & style certain parts of an element or group of elements based on their state position or other characteristics that cannot be achieved by simple element selector

1) :hover:

* Purpose:

* Select & styles an element when the mouse pointer is over it

ex:- a:hover {
color: red;
}

2) :active:

* Purpose: Selects & styles an element while it is being activated or clicked

ex: button:active {
background-color = #3498db;
}

3) :focus:

* purpose:

* Select & style an element that has keyboard focus

ex:- input:focus {
border: 2px solid #27ae60;
}

4) :first-child

* Purpose: Select your child element of a parent

5) :nth-child(n):

* purpose:

Select the nth child element of a parent.