**Q.1 What’s Box Model in CSS?**

**Ans:** The **CSS box model** is a container, It can be used as a toolkit for customizing the layout of different elements and contains multiple properties including borders, margins, padding, and the content itself.

* **Content** - The content of the box, where text and images appear
* **Padding** - Clears an area around the content. The padding is transparent
* **Border** - A border that goes around the padding and content
* **Margin** - Clears an area outside the border. The margin is transparent

**Q.2 What are the Different Types of Selectors in CSS & what are the advantages of them?**

Ans: **CSS selectors** are used to select the content you want to style.CSS selectors select HTML elements according to its id, class, type, attribute etc.

There are several different types of selectors in CSS.

1. CSS Element Selector: The element selector selects the HTML element by name.
2. CSS Id Selector: The id selector selects the id attribute of an HTML element to select a specific element.
3. CSS Class Selector: The class selector selects HTML elements with a specific class attribute.
4. CSS Universal Selector: The universal selector is used as a wildcard character. It selects all the elements on the pages.
5. CSS Group Selector: The grouping selector is used to select all the elements with the same style definitions.

Advantages: CSS selectors tend to perform better, faster, and more reliably than XPath in most browsers. They are much shorter and easier to read and understand.

**Q.3 What is VW/VH?**

Ans: VH stands for “viewport height”, which is the viewable screen's height. 100VH would represent 100% of the viewport's height, or the full height of the screen. VW stands for “viewport width”, which is the viewable screen's width.

While PX is primarily used for font sizing, %, VW, and VH are mostly used for margins, padding, spacing, and widths/heights.

**Q.4 What’s difference between Inline, Inline Block and block?**

Ans: **Inline:** The element does not start on a new line and only occupy just the width it requires. You cannot set the width or height.

**Inline block:** It is formatted just like the inline element, where it doesn’t start on a new line but, you can set width and height values.

**Block:** The element will start on a new line and occupy the full width available. And you can set width and height values.

**Q.5 How is Border-box different from Content Box?**

Ans: In the content box model, the content inside of element will have the same dimension as the element. In the border box model, the content's dimension has to subtract the border and padding from the element's dimension.

**Q.6 What’s z-index and how does it Function?**

Ans: z-index is the CSS property that controls the stacking order of overlapping elements on a page.

Function: Z index only works on positioned elements (position:absolute, position:relative, or position:fixed).

**Q.7 What’s Grid & Flex and difference between them?**

Ans: A grid is a collection of horizontal and vertical lines creating a pattern against which we can line up our design elements. They help us to create layouts in which our elements will not jump around or change width as we move from page to page, providing greater consistency on our websites. CSS Grid Layout is a two-dimensional layout system for the web. It lets you lay content out in rows and columns. It has many features that make building complex layouts straightforward.

Flex: The flex property in CSS is shorthand for **flex-grow, flex-shrink,** and flex-basis. It only works on the flex-items, so if the container's item is not a flex-item, the **flex** property will not affect the corresponding item.

**CSS Flexbox vs CSS Grid:**

Flexbox:

* CSS Flexbox is a one-dimensional layout model.
* A Flexbox container can either facilitate laying out things in a row, or lay them out in a column.
* Flexbox cannot intentionally overlap elements or items in a layout.
* Flexbox is basically content based and it listens to the content and adjusts to it.
* Flexbox can be used for scaling, one-sided aligning, and organizing elements within a container.

Grid:

* CSS Grid is a two-dimensional model.
* Grid can facilitate laying out items across and down at once.
* CSS Grid helps you create layouts with overlapping elements.
* Grid operates more on the layout level and it is container based.
* Grid is useful when you want to define a large-scale layout with more complex and subtle designs.

**Q.8 Difference between absolute and relative and sticky and fixed position explain with Example?**

Ans: [**Absolute Position:**](https://www.geeksforgeeks.org/css-positioning-elements/#:~:text=An%20element%20with%20position%3A%20absolute,%20which%20are%20at%20same%20level.) An element with *position: absolute;* will cause it to adjust its position with respect to its parent. If no parent is present, then it uses the document body as parent.

**Syntax**: position: absolute;

**Example**: In this case, we change the position of the main element to position: absolute. We will also give its parent element a relative position so that it does not get positioned relative to the <html> element.

.main-element {

position: absolute;

left: 10px;

bottom: 10px;

background-color: yellow;

padding: 10px;

}

.parent-element {

position: relative;

height: 100px;

padding: 10px;

background-color: #81adc8;

}

.sibling-element {

background: #f2f2f2;

padding: 10px;

border: 1px solid #81adc8;

}

**Relative position:**

position: relative is similar to static in that relatively positioned elements will follow the normal flow of the webpage. But the main difference is that using relative will now unlock the other CSS layout properties.

**Example:**

.main-element {

position: relative;

left: 10px;

bottom: 10px;

background-color: yellow;

padding: 10px;

}

**Sticky:**

position: sticky is a mix of position: relative and position: fixed. It acts like a relatively positioned element until a certain scroll point and then it acts like a fixed element.

Example:

.main-element {

position: sticky;

top: 10px;

background-color: yellow;

padding: 10px;

}

.parent-element {

position: relative;

height: 800px;

padding: 50px 10px;

background-color: #81adc8;

}

### Fixed:

Fixed position elements are similar to absolutely positioned elements. They are also removed from the normal flow of the document. But unlike absolutely positioned element, they are always positioned relative to the <html> element.

One thing to note is that fixed elements are not affected by scrolling. They always stay in the same position on the screen.

Example:

.main-element {

position: fixed;

bottom: 10px;

left: 10px;

background-color: yellow;

padding: 10px;

}

html {

height: 800px;

}