# CSS INTERVIEW Q&A

**1. Difference between flex-direction as row and column.**

This is a CSS property based on the concept of the flexbox. Flex-direction property can be applied to the parent elements and then the child elements of this parent tag will be placed accordingly.

The difference is when we apply flex-direction : row, child elements will be placed one after another in the same row (horizontally) and then the row or x-axis will be considered as the main axis, in this case the y-axis will be known as cross axis.

HTML:-

```
    <div class="container">
        <div class="flex-items"></div>
        <div class="flex-items"></div>
        <div class="flex-items"></div>
    </div>
```

CSS:-

```
.container{
    Display: flex;
    flex-direction: row;
}

.flex-items{
    width: 100px;
    height: 100px;
    background-color: blueviolet;
    margin: 20px 20px:
}
```

Output:-

On the other hand if we apply flex-direction : column places the child elements in column(vertically) means every child element will be placed one after another in the same column, and then the y-axis will be considered as the main axis and x-axis will be known as cross axis.

HTML:-

```
    <div class="container">
        <div class="flex-items"></div>
        <div class="flex-items"></div>
        <div class="flex-items"></div>
    </div>
```

CSS:-

```
.container{
    display: flex;
    flex-direction: column;
}

.flex-items{
    width: 100px;
    height: 100px;
    background-color: blueviolet;
    margin: 20px 20px;
```

Output:-

**********

**2. Explain the Inline, Internal, and External Stylesheet.**

In CSS, we have 3 different ways of writing styles for HTML elements

- Internal
- Inline
- External
- Inline styling gets applied directly to the element in the style attribute. This styling is having the highest priority among all. But it is not advised to use this styling much in projects.

HTML:-

```
<h1 style="color: red; text-align:center;">Heading One</h1>
<p style="color: blue;">This is a paragraph</p>
```

Output:-

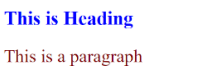

Internal styling gets written in the head tag of the HTML file. It will be written in the style tag. The styling written in style tag will be applicable only in that HTML file.

HTML:-

```
<head>
        <style>
        h1 {
          color: maroon;
          margin-left: 40px;
        }


          p{
                color: maroon;
        }
        </style>
        </head>
</head>
<body>
    <h1>This is  Heading</h1>
    <p>This is a paragraph</p>
</body>
```

Output:



External Styling is done by creating a different CSS file with .css extension. This file can be linked to any HTML file wherever the styling is required. The external CSS file can be attached to the HTML file in the head tag in the link tag. This styling is having the least priority but it is the most advised way of writing styling in projects.

HTML:-

```
<head>
    <title>Document</title>
    <link rel="stylesheet" href="mystyle.css">
</head>
<body>
    <h1>This is a heading</h1>
    <p>This is a paragraph.</p>
</body>
```

CSS:- (file name : mystyle.css)

```
h1 {
    color: blue;
    margin-top: 40px;
  }
p{
    color: maroon;
    font-size: 30px;
}
```

Output:-

**This is Heading**

This is a paragraph

**3. What does justify-content allow you to do?**

The justify-content property is a sub-property of the Flex-box Module.

The justify-content property specifies how flex-items are distributed along the main axis of their parent flex-container. If the flex-direction of the parent container is row(default value) then the main axis will be the x-axis. If the flex-direction is a column then the main axis will be the y-axis. This property has 6 different possible values:

- flex-start (default)
- flex-end
- center
- space-around
- space-between
- space-evenly
- stretch

**4. Difference between Absolute and Relative Positioning?**

position: relative

- It is used when we need to position the HTML element relative to its normal position.
- We can set top, right, bottom and left properties that will cause the element to adjust away from the normal position.

HTML:-

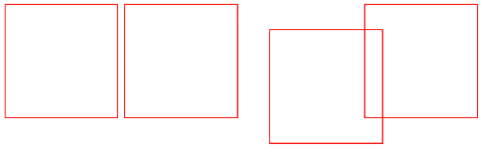
```
<div class="box" id="box1"></div>
```

```
    <div class="box" id="box2"></div>
    <div class="box" id="box3"></div>
  <div class="box" id="box4"></div>
```

CSS:-

```
.box {
   border: 2px solid red;
   display: inline-block;
   width: 150px;
   height: 150px;
   margin: 2px;
  }
 #box3 {
   position: relative;
   top: 34px;
   left: 34px;
 }
```

Output:-



position: absolute

- An element with position absolute will move according to the position of its parent element.
- In the absence of the parent element, the HTML element will be placed according to the body tag. In this case if you want to place the element at any position based on the parent element then we need to give the position : relative to parent element and then position : absolute to child element.

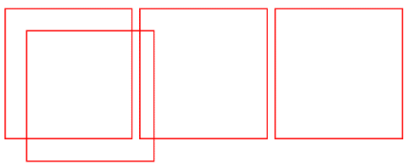
HTML:-

```
    <div class="box" id="box1"></div>
    <div class="box" id="box2"></div>
    <div class="box" id="box3"></div>
  <div class="box" id="box4"></div>
```

CSS:-

```
.box {
   border: 2px solid red;
   display: inline-block;
   width: 150px;
   height: 150px;
   margin: 2px;
  }
 #box3 {
   position: absolute;
   top: 34px;
   left: 34px;
 }
```

Output:-



**5. What are grid-template-columns used for?**

Grid-template-columns property defines how many columns should be there in a **grid-container**. The number of columns is determined by the number of **values** defined in the space-separated list. It takes values in px, % and fraction **(fr).**

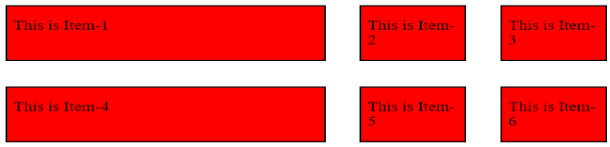
HTML:-

```
<div class="container">
        <div class="item">This is Item-1</div>
        <div class="item">This is Item-2</div>
        <div class="item">This is Item-3</div>
        <div class="item">This is Item-4</div>
        <div class="item">This is Item-5</div>
        <div class="item">This is Item-6</div>
</div>
```

CSS:-

```
.container{
    display: grid;
    grid-template-columns: 300px 100px 100px;     //using pixels
    grid-gap: 2rem;
}
.item{
    background-color: red;
    border: 2px solid black;
    padding: 15px 5px;
}
```

Output:-




CSS:-

```
.container{
    display: grid;
     grid-template-columns: 300px auto 100px;     //using value auto
    grid-gap: 2rem;
}
.item{
    background-color: red;
    border: 2px solid black;
    padding: 15px 5px;
}
```

Output:-

CSS:-

```
.container{
    display: grid;
     grid-template-columns: 1fr 3fr 1fr;          //using fraction
    grid-gap: 2rem;
}
.item{
    background-color: red;
```

```
    border: 2px solid black;
    padding: 15px 5px;
}
```

Output:-


CSS:-

```
.container{
    display: grid;
     grid-template-columns: repeat(3, 25%);       //using percentage and repeat
    grid-gap: 2rem;
}
.item{
    background-color: red;
    border: 2px solid black;
    padding: 15px 5px;
}
```

Output:-


**6. What is z-index in CSS?**

The z-index property determines the stack level of an HTML element. The "stack level" refers to the element's position on the *Z axis* (as opposed to the *X axis* or *Y* axis). Elements with a higher index will be placed on top of elements with a lower index. If the z-index value of any element is positive then that element will be at the top and element having negative value of z-index will be at the bottom.

Note :- z-index will only work on an element whose position property has been set to absolute, relative and fixed.

HTML:-

```
<div id="box1"></div>
<div id="box2"></div>
```

CSS:-

```
#box1 {
    width: 100px;
    height: 100px;
    top: 69px;
    position: relative;
    background-color: greenyellow;
    z-index: 1;
}
#box2 {
    width: 150px;
    height: 150px;
    top: 34px;
    left: 20px;
    position: absolute;
    background-color: rebeccapurple;
```

```
    z-index: 0;
}
```

Output:-

**7. What is the difference between Padding and Margin.**

These two properties are CSS properties that provide space or gap. Both margin and padding targets the four sides of an element.

| Margin | Padding |
| --- | --- |
| Margin provides the space between the border and nearby elements. It is the space outside the border. | Padding provides the space between the border and the content of an element. It is the space inside the border |
| You can set margin values to be negative if you want elements to overlap. | Padding values can only be positive. |
| When you set margin values, the element's size does not change, only the space around it. | When you set padding values, the size of that element increases. |

Illustration:-