Additional Notes

Z-index

- Z-index property in CSS is used to specify the stacking order of the positioned elements (elements whose position value is either fixed, absolute, relative, or sticky).
- Z-index only affects positioned elements. Setting a z-index on an unpositioned element does nothing.

Syntax:

z-index : auto | number;

Possible values

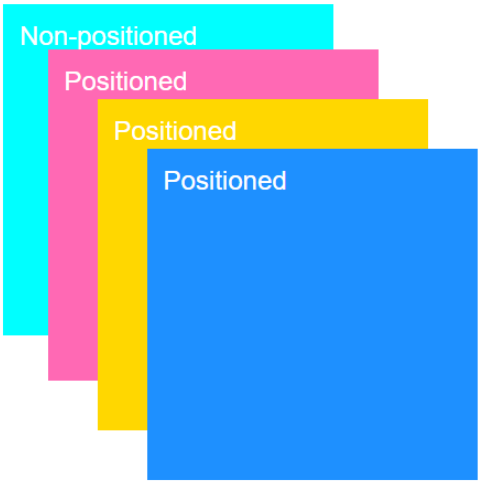
- **auto:** (default) It means that the order of the stack is equivalent to the parent.
- **number:** It means that the element's stack level is set to the given value. It also allows negative values.
  E.g. When no Z-index is applied.
  HTML:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Z-index Demonstration</title>
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <div class="pink">Positioned
        <div class="orange">Positioned</div>
    </div>
    <div class="blue">Positioned</div>
    <div class="cyan">Non-positioned</div>
</body>
</html>
```

CSS:

```css
div {
    width: 200px;
    height: 200px;
    padding: 10px;
    box-sizing: border-box;
    font-family: Arial, sans-serif;
    color: white;
}
.pink, .blue, .orange {
    position: absolute;
}
.pink {
    background-color: #ff69b4;
    top: 35px;
    left: 35px;
}
.orange {
    background-color: #ffd700;
    top: 30px;
    left: 30px;
}
.blue {
    background-color: #1e90ff;
    top: 95px;
    left: 95px;
}
.cyan {
    background-color: #00ffff;
}
```

Output:



In this case, the stacking order is determined by the HTML structure. The cyan div appears on top because it's last in the HTML and not positioned.

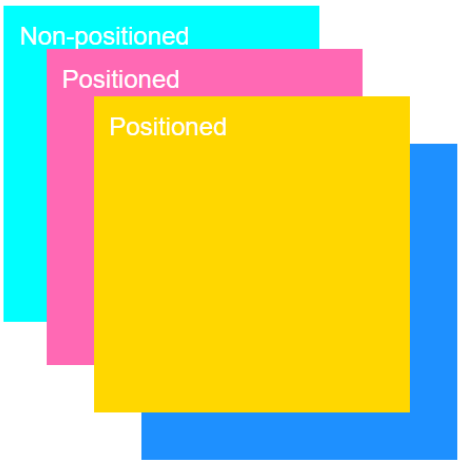
Suppose you want to change the stacking order of these elements by using the z-index property.

- An element with a higher z-index will be displayed in front of an element with a lower z-index.

Apply this styling:

```
div {
    width: 200px;
    height: 200px;
    padding: 10px;
    box-sizing: border-box;
    font-family: Arial, sans-serif;
    color: white;
}
.pink, .blue, .orange {
    position: absolute;
}
.pink {
    background-color: #ff69b4;
    top: 35px;
    left: 35px;
}
.orange {
    background-color: #ffd700;
    top: 30px;
    left: 30px;
    z-index: 3;
}
.blue {
    background-color: #1e90ff;
    top: 95px;
    left: 95px;
    z-index: 2;
}
.cyan {
    background-color: #00ffff;
    z-index: 1;
}
```

Output:



In this case, the z-index property controls the stacking order:

- The orange div (z-index: 3) appears on top
- The blue div (z-index: 2) is next
- The cyan div (z-index: 1) is at the bottom
- The pink div has no specified z-index, so it's treated as if it had z-index: auto, placing it below elements with positive z-index

Transforms

- The transform property allows you to manipulate an element visually.
- It has the following methods: rotate(),translate(), scale(), move(), skew() and matrix().

rotate():

- By using the rotate() method, you can rotate an element clockwise or anticlockwise from its current position.
- A positive value will rotate it in a clockwise direction while a negative value will rotate it in the anti-clockwise direction.

e.g. if you want to rotate an element in a clockwise direction with 90 degrees:

```
div {
  transform: rotate(90deg);
}
```

Note: you will be learning transform property in detail in the upcoming lecture.