Position in Css

CSS Positioning

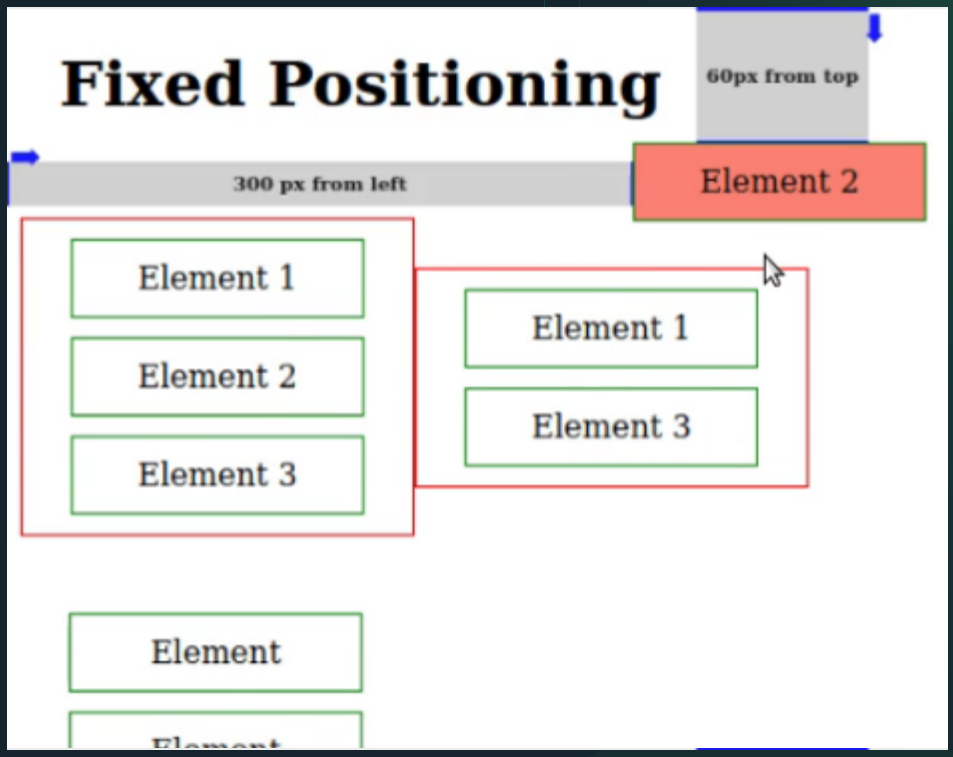
Static Positioning

Static positioning is the default positioning behaviour of elements. Elements with static positioning are positioned according to the normal flow of the document. In other words, they are not affected by any positioning properties.

Example

```
<!DOCTYPE html>
<html>
<head>
  <style>
    .box {
      width: 200px;
      height: 200px;
      background-color: red;
    }
  </style>
</head>
<body>
  <div class="box">Static Positioning</div>
</body>
</html>
```

Fixed Positioning



Fixed positioning allows you to position an element relative to the browser window, regardless of scrolling. The element will stay in the same position even if the page is scrolled.

Example

```html
<!DOCTYPE html>
<html>
<head>
  <style>
    .box {
      position: fixed;
      top: 50px;
      left: 50px;
      width: 200px;
      height: 200px;
      background-color: blue;
    }
  </style>
</head>
<body>
  <div class="box">Fixed Positioning</div>
</body>
</html>
```

Sticky Positioning

Sticky positioning is a hybrid between relative and fixed positioning. The element is initially positioned according to the normal flow of the document, but it becomes fixed once it reaches a specified threshold (usually when it hits the top of the viewport while scrolling).

Example

```
<!DOCTYPE html>
<html>
<head>
  <style>
    .box {
      position: sticky;
      top: 50px;
      width: 200px;
      height: 200px;
      background-color: green;
    }
  </style>
</head>
<body>
  <div class="box">Sticky Positioning</div>
  <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque viverra semper lectus, non tincidunt purus facilisis ac.</p>
</body>
</html>
```

Relative Positioning

Relative positioning allows you to position an element relative to its normal position in the document flow. It does not affect the positioning of other elements on the page.

Example

```
<!DOCTYPE html>
<html>
<head>
  <style>
    .box {
      position: relative;
      left: 50px;
      top: 50px;
      width: 200px;
      height: 200px;
      background-color: yellow;
    }
  </style>
</head>
<body>
  <div class="box">Relative Positioning</div>
</body>
</html>
```

Absolute Positioning

Absolute positioning allows you to position an element relative to its closest positioned ancestor (an ancestor with a position other than static) or the document itself. The element is taken out of the normal document flow and other elements ignore it.

Example

```
<!DOCTYPE html>
<html>
<head>
  <style>
    .container {
      position: relative;
      width: 400px;
      height: 300px;
      background-color: lightgray;
    }

    .box {
      position: absolute;
      top: 50px;
      left: 50px;
      width: 200px;
      height: 200px;
      background-color: orange;
    }
  </style>
</head>
<body>
  <div class="container">
    <div class="box">Absolute Positioning</div>
  </div>
</body>
</html>
```

CSS Z-Index Layer

CSS layers are created using the z-index property, which controls the vertical stacking order of elements on a web page. The z-index property accepts integer values, where a higher value places an element on top of elements with lower values.

Example

```html
<!DOCTYPE html>
<html>
<head>
  <style>
    .layer1 {
      position: absolute;
      top: 50px;
      left: 50px;
      width: 200px;
      height: 200px;
      background-color: red;
      z-index: 1;
    }

    .layer2 {
      position: absolute;
      top: 100px;
      left: 100px;
      width: 200px;
      height: 200px;
      background-color: blue;
      z-index: 2;
    }

    .layer3 {
      position: absolute;
      top: 150px;
      left: 150px;
      width: 200px;
      height: 200px;
      background-color: green;
      z-index: 3;
    }
  </style>
</head>
```

- In the above example, we have three elements with different z-index values. The element with the class layer3 has the highest z-index and will be displayed on top of the other elements. The element with the class layer2 will be displayed above the element with the class layer1 but below the element with the class layer3.
- By adjusting the z-index values, you can control the stacking order of elements and create layered layouts in your web page. It's important to note that the z-index property only works on positioned elements (i.e., elements with a position value other than static).
- Remember that when using z-index, elements must have a position other than static (e.g., relative, absolute, or fixed) for the z-index property to have an effect.

CSS Overflow

The CSS overflow property is used to control how content that exceeds the dimensions of an element is displayed. It determines whether to clip, scroll, or display the overflowed content. There are four possible values for the overflow property:

Visible (default)

This value allows the content to overflow the boundaries of the element. It may overlap with other elements.

Example

```
<!DOCTYPE html>
<html>
<head>
  <style>
    .box {
      width: 200px;
      height: 100px;
      background-color: lightblue;
      overflow: visible;
    }
  </style>
</head>
<body>
  <div class="box">
    Lorem ipsum dolor sit amet, consectetur
adipiscing elit.
  </div>
</body>
</html>
```

Hidden

This value clips the overflowed content and hides it. The content that exceeds the element's dimensions will not be visible.

Example

```
<!DOCTYPE html>
<html>
<head>
  <style>
    .box {
      width: 200px;
      height: 100px;
      background-color: lightblue;
      overflow: hidden;
    }
  </style>
</head>
<body>
  <div class="box">
    Lorem ipsum dolor sit amet, consectetur
adipiscing elit.
  </div>
</body>
</html>
```

Scroll

This value adds scrollbars to the element, allowing the user to scroll and view the overflowed content. Scrollbars are added regardless of whether the content exceeds the element's dimensions.

Example

```
<!DOCTYPE html>
<html>
<head>
  <style>
    .box {
      width: 200px;
      height: 100px;
      background-color: lightblue;
      overflow: scroll;
    }
  </style>
</head>
<body>
  <div class="box">
    Lorem ipsum dolor sit amet, consectetur
adipiscing elit.
  </div>
</body>
</html>
```

Auto

This value adds scrollbars to the element only if the content exceeds the element's dimensions. If the content fits within the element, no scrollbars will be displayed.

Example

```
<!DOCTYPE html>
<html>
<head>
  <style>
    .box {
      width: 200px;
      height: 100px;
      background-color: lightblue;
      overflow: auto;
    }
  </style>
</head>
<body>
  <div class="box">
    Lorem ipsum dolor sit amet, consectetur
adipiscing elit.
  </div>
</body>
</html>
```

The overflow property is useful when dealing with elements that have limited dimensions and need to handle content that exceeds those dimensions. By setting the appropriate overflow value, you can control how the overflowed content is displayed to the user.

CSS Psuedo Element

CSS pseudo-elements allow you to style and target specific parts of an element's content, such as the first letter, first line, or specific parts of a list. Pseudo-elements are denoted by double colons (::) and are used in conjunction with selectors to apply styles to specific parts of the content.

Here are a few commonly used CSS pseudo-elements: