Layout in CSS

Display Property:

The `display` property in CSS is crucial for determining how an element is rendered on the webpage. Different values of the `display` property result in different layouts. The commonly used values are:

1. Block:
- Elements with `display: block` generate a block-level box.
- These elements typically start on a new line and take up the full width available, extending from the left edge to the right edge of their containing element.
- Examples include `<div>`, `<p>`, and `<h1>`.

```
div {
  display: block;
}
```

2. Inline:
- Elements with `display: inline` generate an inline-level box.
- These elements do not start on a new line and only take up as much width as necessary. They flow within the content, side by side.
- Examples include `<span>`, `<a>`, and `<img>`.

```
span {
  display: inline;
}
```

3. Inline-Block:
- Elements with `display: inline-block` combine features of both block and inline elements.
- They remain in the flow like inline elements but can have block-level properties like width and height.
- Useful for creating elements that need to be inline but also have block-level styling.

```
button {
  display: inline-block;
}
```

Positioning:

The `position` property in CSS is used to control the positioning of an element. The values include:

1. Relative:
- Elements with `position: relative` are positioned relative to their normal position in the document flow.
- Shifting an element using `top`, `right`, `bottom`, or `left` will move it from its default position.

```
div {
  position: relative;
  top: 10px;
  left: 20px;
}
```

2. Absolute:
- Elements with `position: absolute` are positioned relative to the nearest positioned ancestor (an ancestor with a position other than static) instead of the normal flow.
- If no positioned ancestor is found, it positions relative to the initial containing block.

```
div {
  position: absolute;
  top: 50px;
  left: 100px;
}
```

3. Fixed:
- Elements with `position: fixed` are positioned relative to the browser window or the viewport.

- They remain fixed even when the page is scrolled.

```
header {
  position: fixed;
  top: 0;
  left: 0;
  width: 100%;
}
```

Floats and Clear:

1. Floats:
- The `float` property is used to push an element to one side and make it float.
- Elements can be floated left or right, and text or other elements will wrap around it.
- Commonly used for creating simple layouts, but it has some drawbacks and is less recommended in modern CSS layout techniques.

```
img {
  float: left;
  margin-right: 10px;
}
```

2. Clear:
- The `clear` property is used to control the behavior of elements adjacent to floated elements.
- It specifies on which sides of an element floating elements are not allowed to float.

```
article {
  clear: both;
}
```

These layout techniques are fundamental in web development and are often used in combination to create complex and responsive designs. As the web evolves, new layout techniques (e.g., Flexbox and Grid) have been introduced, offering more powerful and flexible options for creating layouts.