Here's the Leatest notes on the `display` properties in CSS:

Display Properties in CSS

The `display` property in CSS defines how an element is displayed on the web page. It's one of the most important properties for layout and design.

Common Display Values

1. `block`

- The element takes up the full width available and starts on a new line.
- Examples: `<div>`, `<h1>`–`<h6>`, `<p>`, `<header>`, `<footer>`, `<section>`.

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

2. `inline`

- The element only takes up as much width as necessary and does not start on a new line.
- Examples: `<span>`, `<a>`, `<img>`, `<strong>`, `<em>`.

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

3. `inline-block`

- The element flows like an inline element but can have a width and height set like a block element.
- It does not start on a new line.

```
img {
    display: inline-block;
    width: 100px;
    height: 100px;
}
```

4. `none`

- The element is completely removed from the document flow and will not be displayed on the page. No space is reserved for it.

```
.hidden {
    display: none;
}
```

5. `flex`

- Enables a flexbox layout, allowing for flexible and responsive designs.
- Parent element becomes a flex container, and its children become flex items.

```
.container {
    display: flex;
}
```

6. `grid`

- Enables a grid layout, allowing for a two-dimensional layout system with rows and columns.

```
.container {
    display: grid;
}
```

7. `inline-flex`

- Behaves like `inline-block` but for a flex container.

```
.container {
    display: inline-flex;
}
```

8. `inline-grid`

- Behaves like `inline-block` but for a grid container.

```
.container {
    display: inline-grid;
}
```

9. `table`

- The element behaves like a `<table>` element, with table-row and table-cell behavior.

```
.container {
    display: table;
}
```

10. `table-row`, `table-cell`, etc.

- These values make an element behave like `<tr>` (table row), `<td>` (table cell), and other table-related elements.

```
.row {
    display: table-row;
}
.cell {
    display: table-cell;
}
```

Usage and Examples

Block-Level Element Example

```
<div style="display: block;">
    This is a block-level element.
</div>
```

Inline-Level Element Example

```
<span style="display: inline;">
    This is an inline element.
</span>
```

Inline-Block Example

```
<div style="display: inline-block; width: 200px;">
    This is an inline-block element.
</div>
```

Flexbox Example

```
<div class="container" style="display: flex;">
    <div>Flex Item 1</div>
```

```
    <div>Flex Item 2</div>
    <div>Flex Item 3</div>
</div>
```

Grid Example

```
<div class="container" style="display: grid;">
    <div>Grid Item 1</div>
    <div>Grid Item 2</div>
    <div>Grid Item 3</div>
</div>
```

Key Points to Remember

- `block` elements take up the full width of their parent container.
- `inline` elements only take up as much space as needed.
- `inline-block` elements combine the characteristics of both `block` and `inline` elements.
- `none` hides the element entirely from the page.
- `flex` and `grid` are powerful tools for creating responsive layouts.