Chapter 23: Layout Control

Value	Effect
`none`	Hide the element and prevent it from occupying space.
`block`	Block element, occupy 100% of the available width, break after element.
`inline`	Inline element, occupy no width, no break after element.
`inline-block`	Taking special properties from both inline and block elements, no break, but can have width.
`inline-flex`	Displays an element as an inline-level flex container.
`inline-table`	The element is displayed as an inline-level table.
`grid`	Behaves like a block element and lays out its content according to the grid model.
`flex`	Behaves like a block element and lays out its content according to the flexbox model.
`inherit`	Inherit the value from the parent element.
`initial`	Reset the value to the default value taken from behaviors described in the HTML specifications or from the browser/user default stylesheet.
`table`	Behaves like the HTML `table` element.
`table-cell`	Let the element behave like a **`<td>`** element
`table-column`	Let the element behave like a **`<col>`** element
`table-row`	Let the element behave like a **`<tr>`** element
`list-item`	Let the element behave like a **`<li>`** element.

Section 23.1: The display property

The `display` CSS property is fundamental for controlling the layout and flow of an HTML document. Most elements have a default `display` value of either `block` or `inline` (though some elements have other default values).

Inline

An `inline` element occupies only as much width as necessary. It stacks horizontally with other elements of the same type and may not contain other non-inline elements.

```
<span>This is some <b>bolded</b> text!</span>
```

This is some **bolded** text!

As demonstrated above, two `inline` elements, **`<span>`** and **`<b>`**, are in-line (hence the name) and do not break the flow of the text.

Block

A `block` element occupies the maximum available width of its' parent element. It starts with a new line and, in contrast to `inline` elements, it does not restrict the type of elements it may contain.

```
<div>Hello world!</div><div>This is an example!</div>
```

Hello world!
This is an example!

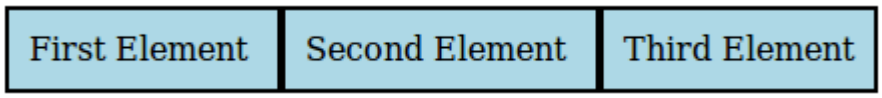
The `div` element is block-level by default, and as shown above, the two `block` elements are vertically stacked and, unlike the `inline` elements, the flow of the text breaks.

Inline Block

The `inline-block` value gives us the best of both worlds: it blends the element in with the flow of the text while allowing us to use `padding`, `margin`, `height` and similar properties which have no visible effect on `inline` elements.

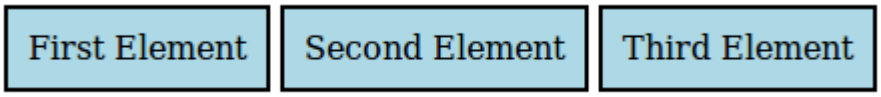
Elements with this display value act as if they were regular text and as a result are affected by rules controlling the flow of text such as `text-align`. By default they are also shrunk to the the smallest size possible to accommodate their content.

```
<!--Inline: unordered list-->
<style>
li {
    display : inline;
    background : lightblue;
    padding:10px;

    border-width:2px;
    border-color:black;
    border-style:solid;
    }
</style>

<ul>
<li>First Element </li>
<li>Second Element </li>
<li>Third Element </li>
</ul>
```




```
<!--block: unordered list-->
<style>
li {
    display : block;
    background : lightblue;
    padding:10px;

    border-width:2px;
    border-color:black;
    border-style:solid;
    }
</style>

<ul>
<li>First Element </li>
<li>Second Element </li>
<li>Third Element </li>
</ul>
```

```
<!--Inline-block: unordered list-->
<style>
li {
    display : inline-block;
    background : lightblue;
    padding:10px;

    border-width:2px;
    border-color:black;
    border-style:solid;
    }

</style>

<ul>
<li>First Element </li>
<li>Second Element </li>
<li>Third Element </li>
</ul>
```



none

An element that is given the none value to its display property will not be displayed at all.

For example let's create a div-element that has an id of `myDiv`:

```
<div id="myDiv"></div>
```

This can now be marked as not being displayed by the following CSS rule:

```
#myDiv {
    display: none;
}
```

When an element has been set to be `display:none;` the browser ignores every other layout property for that specific element (both `position` and `float`). No box will be rendered for that element and its existence in html does not affect the position of following elements.

Note that this is different from setting the `visibility` property to `hidden`. Setting `visibility: hidden;` for an element would not display the element on the page but the element would still take up the space in the rendering process as if it would be visible. This will therefore affect how following elements are displayed on the page.

The `none` value for the display property is commonly used along with JavaScript to show or hide elements at will, eliminating the need to actually delete and re-create them.

Section 23.2: To get old table structure using div

This is the normal HTML table structure

```
<style>
    table {
        width: 100%;
    }
```

```
</style>

<table>
  <tr>
    <td>
        I'm a table
    </td>
  </tr>
</table>
```

You can do same implementation like this

```
<style>
    .table-div {
        display: table;
    }
    .table-row-div {
        display: table-row;
    }
    .table-cell-div {
        display: table-cell;
    }
</style>

<div class="table-div">
  <div class="table-row-div">
    <div class="table-cell-div">
      I behave like a table now
    </div>
  </div>
</div>
```