CSS Selectors

Module 3 : 02

Objectives

1. HTML Document Hierarchy
2. Including CSS
3. CSS Selectors
4. Cascade
5. Box Model
6. Display
7. Positioning
8. Floats
9. Units of Measurement

HTML Element Hierarchy

Each HTML element has a hierarchical relationship with other elements.

1. All elements, except the root element (<html>) has a single parent element

   A Parent is the element that the current element is included in.

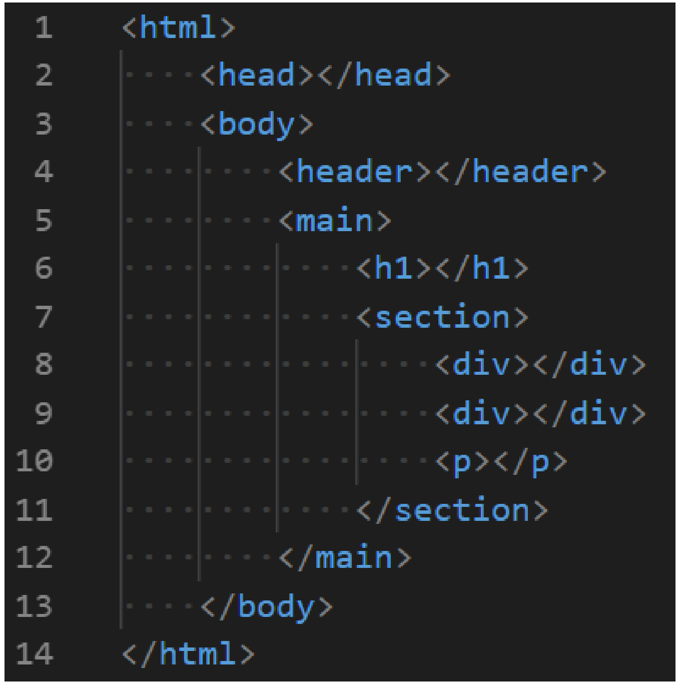
1. Elements can have 0...n child elements

   Children are elements included in the current element.

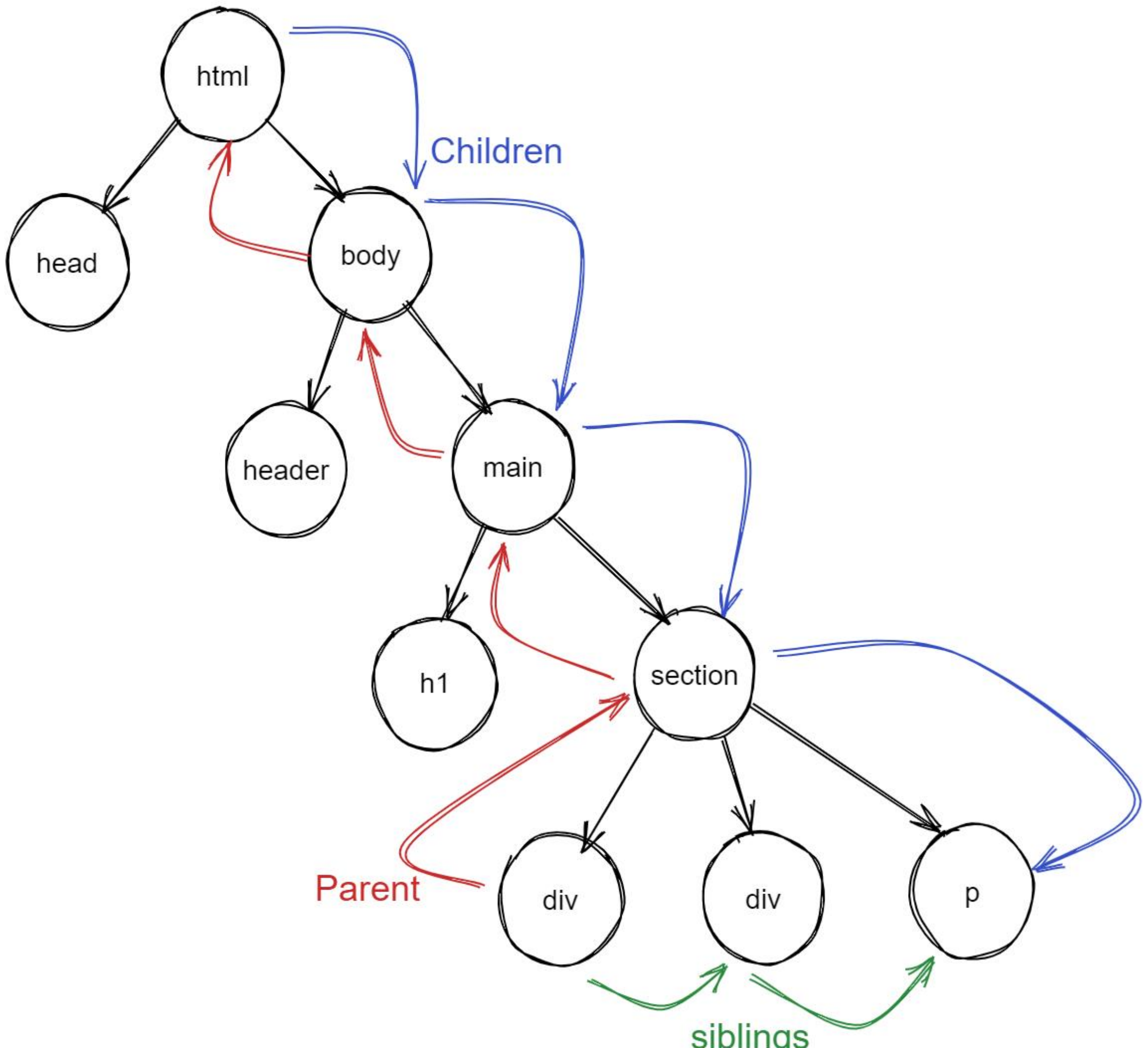
1. Elements can have 0...n sibling elements

   Sibling elements are elements with the same parent

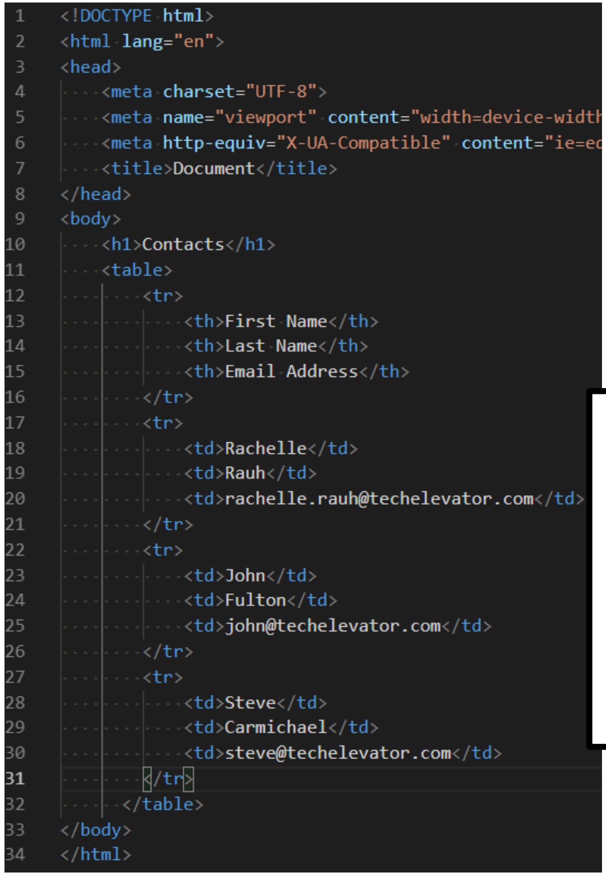
Element Hierarchy



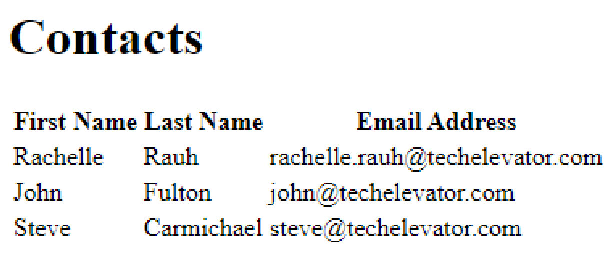
```
<html>
    <head></head>
    <body>
        <header></header>
        <main>
            <h1></h1>
            <section>
                <div></div>
                <div></div>
                <p></p>
            </section>
        </main>
    </body>
</html>
```



Document Object Model (DOM)

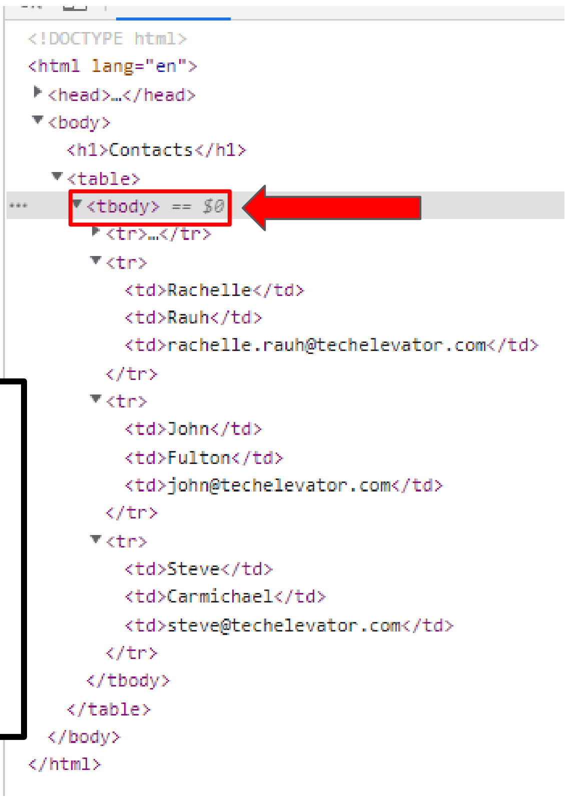


```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width
    <meta http-equiv="X-UA-Compatible" content="ie=ed
    <title>Document</title>
</head>
<body>
    <h1>Contacts</h1>
    <table>
        <tr>
            <th>First Name</th>
            <th>Last Name</th>
            <th>Email Address</th>
        </tr>
        <tr>
            <td>Rachelle</td>
            <td>Rauh</td>
            <td>rachelle.rauh@techelevator.com</td>
        </tr>
        <tr>
            <td>John</td>
            <td>Fulton</td>
            <td>john@techelevator.com</td>
        </tr>
        <tr>
            <td>Steve</td>
            <td>Carmichael</td>
            <td>steve@techelevator.com</td>
        </tr>
      </table>
</body>
</html>
```



Contacts

First Name	Last Name	Email Address
Rachelle	Rauh	rachelle.rauh@techelevator.com
John	Fulton	john@techelevator.com
Steve	Carmichael	steve@techelevator.com



```
<!DOCTYPE html>
<html lang="en">
 ▸<head>…</head>
 ▾<body>
    <h1>Contacts</h1>
   ▾<table>
...  ▾<tbody> == $0
      ▸<tr>…</tr>
      ▾<tr>
         <td>Rachelle</td>
         <td>Rauh</td>
         <td>rachelle.rauh@techelevator.com</td>
       </tr>
      ▾<tr>
         <td>John</td>
         <td>Fulton</td>
         <td>john@techelevator.com</td>
       </tr>
      ▾<tr>
         <td>Steve</td>
         <td>Carmichael</td>
         <td>steve@techelevator.com</td>
       </tr>
     </tbody>
   </table>
 </body>
</html>
```

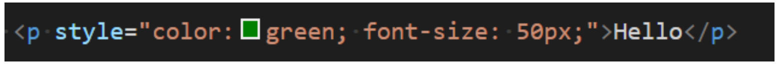
In the html, there is a <table> element with headers, rows and cells. When Chrome renders the page, it creates the DOM (seen in the inspector tool).

Note how the DOM includes an extra <tbody> element that is not in the html.

Including CSS

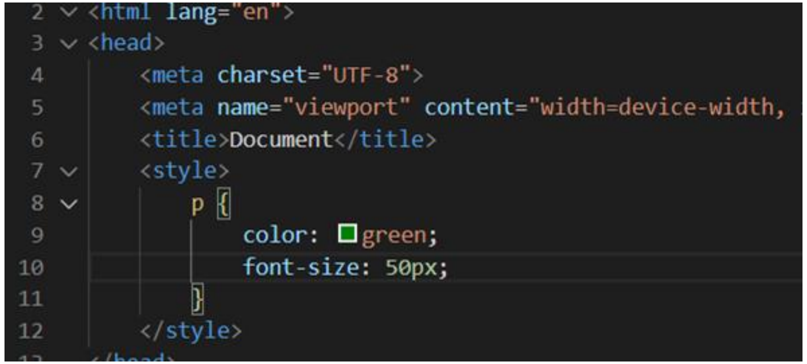
CSS Can be included in HTML in 3 ways

1. inline on an HTML tag using the *style* attribute



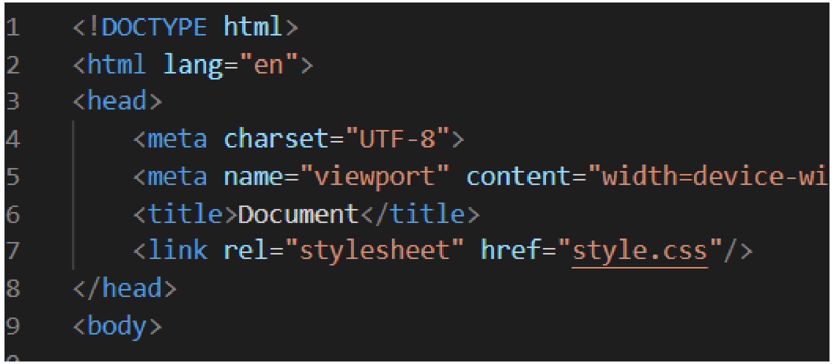
```
<p style="color: green; font-size: 50px;">Hello</p>
```

2. Within the head of the document in a <style></style> element



```
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
    <title>Document</title>
    <style>
        p {
            color: green;
            font-size: 50px;
        }
    </style>
```

3. As a link to an external CSS document **(Preferred method)**



```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-wi
    <title>Document</title>
    <link rel="stylesheet" href="style.css"/>
</head>
<body>
```

CSS Selectors

Universal Selector:	*****	**Selects all elements in the document** *** {font-size: 30px;}**
Element Selector:	**div**	**Selects ALL elements of that type** **<div>**
Class Selector:	**.spanOne**	**Selects all elements with that class as an attribute** **<span class=”spanOne”>**
Id Selector:	**#divThree**	**Selects the element with that id as an attribute** **<div id=”divThree”>**

List of Pseudo-classes Selector CodePen

CSS Selectors

CSS Selectors can be combined to achieve greater specificity

```
        div h2.header ul > li {
                background-color: red;
    }
```

This selector will select all **li** elements that are a child of an **ul** that is a descendent of an h2 element that has the **.header** class applied to it, that is a descendent of a **div**

```
<div>
    <h2 >
        <ul>
            <li></li>
            <li></li>
        </ul>
    </h2>
</div>

<div>
    <h2 class="header">
        <ul>
            <li></li>
            <li></li>
        </ul>
    </h2>
</div>
```

CSS Selectors continued...

1. Universal Selector:	*****
2. Element Selector:	**div**
3. Class Selector:	**.spanOne**
4. Id Selector:	**#divThree**

---

5. Descendent Combinator:	**.special h2**
6. Child Selector:	**#divThree > span**
7. Adjacent Sibling Selector:	**.spanOne + span**
○ (aka immediately following)	
8. General Sibling Selector:	**.spanTwo ~ span**
9. Pseudo-class Selector:	**div :first-child**
	div :last-child
	div :nth-child(2)
	div:hover

Advanced Selectors: Pseudo-class

Example

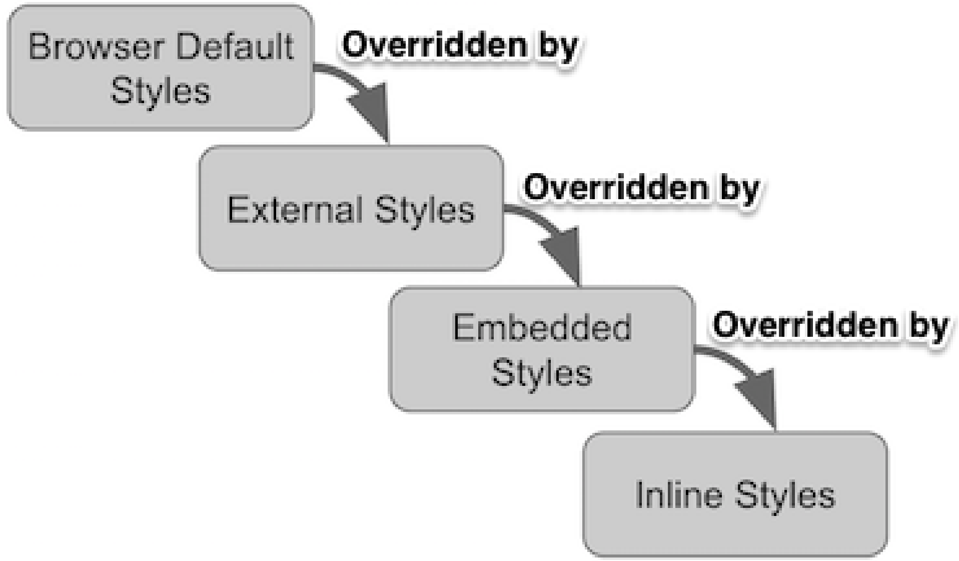
```
/* Any button over which the user's pointer is hovering */
button:hover {
  color: blue;
}
```

Keyword added to selector that specifies a special state of the element.

```
/* unvisited link */
a:link {
  color: #FF0000;
}

/* visited link */
a:visited {
  color: #00FF00;
}

/* mouse over link */
a:hover {
  color: #FF00FF;
}

/* selected link */
a:active {
  color: #0000FF;
}
```

Cascade

Cascade: All CSS applied to an element, including from multiple style sheets, are brought together and then the most specific applied.



Cascading Order

Highest Priority

1. Inline Styles
2. Internal Styles
3. External Styles
4. Browser defaults

Last Rule

- If two selectors are identical, the later of the two will be applied

Specific

- The most specific selector will be applied. h1 {} is more specific than * {}

!Important

- indicates that this rule is more important than previous rules that may be applied.

What is !important?

The !important rule in CSS is used to add more importance to a property/value than normal.

In fact, if you use the !important rule, it will override ALL previous styling rules for that specific property on that element!

```
#myid {
  background-color: blue;
}

.myclass {
  background-color: gray;
}

p {
  background-color: red !important;
}
```

Box Model

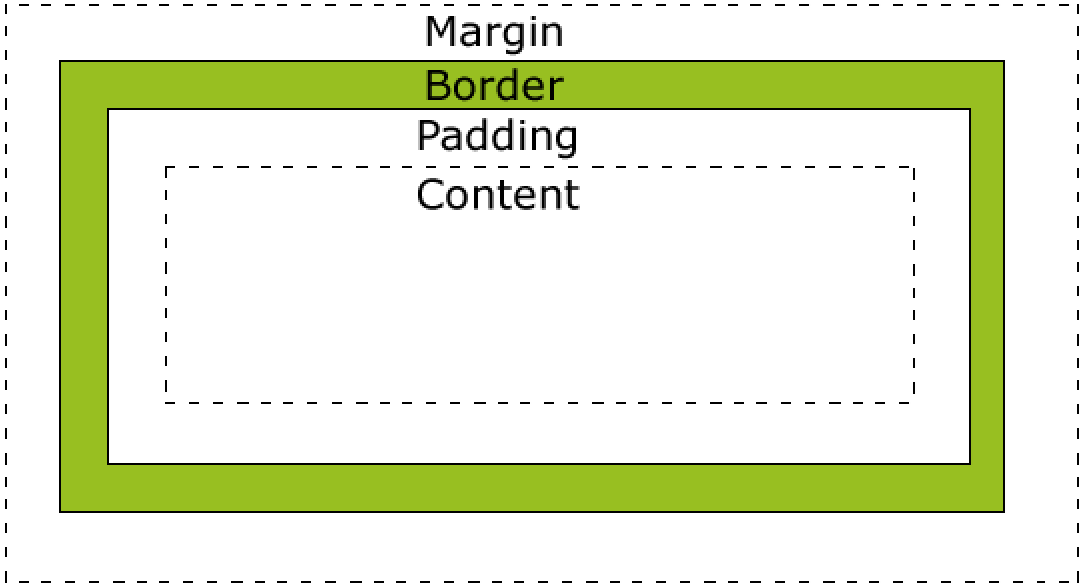
All elements are represented on the page as a box with 4 parts.

1. **content**

2. **padding** - space between the content and the border

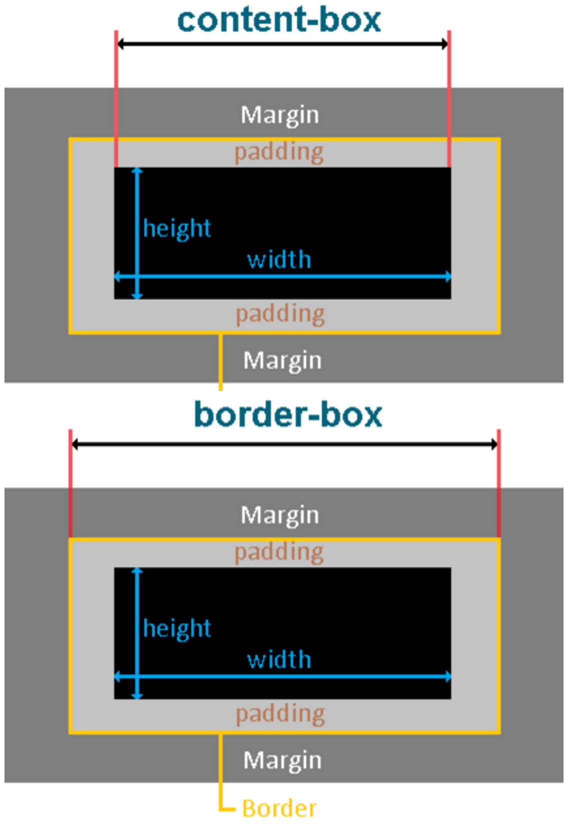
3. **border** - border around the box

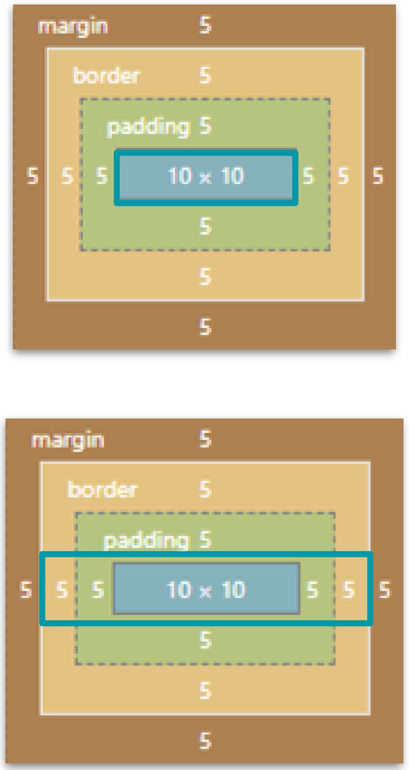
4. **margin** - space outside the border before another box can start



width: 100% is not 100% the size of the content. It is 100% of the space allowed inside the container box the element box is within.

BOX-SIZING





content-box - **default**, width/height include only the content.

border-box - the width/height includes content, padding, and border.

```
* {
  box-sizing: border-box;
}
```

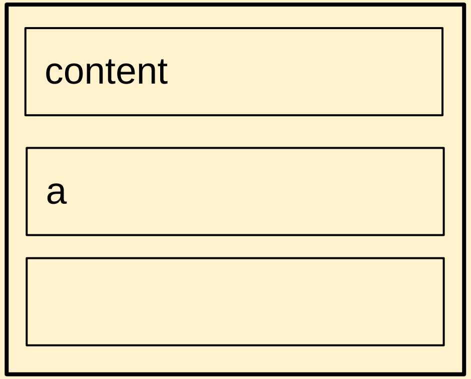
[Box Sizing on W3Schools]

Display

block

1. Starts on a newline
2. Takes up full available width
3. can set width/ height

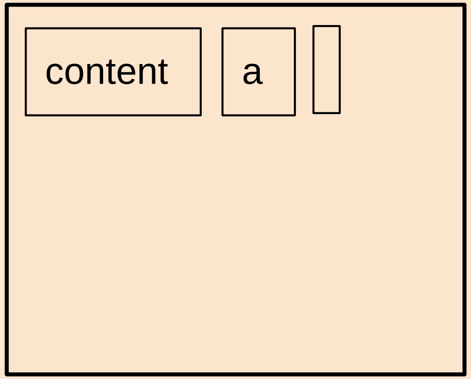
default for: div, h1, p, form, header, footer, section, etc.



inline

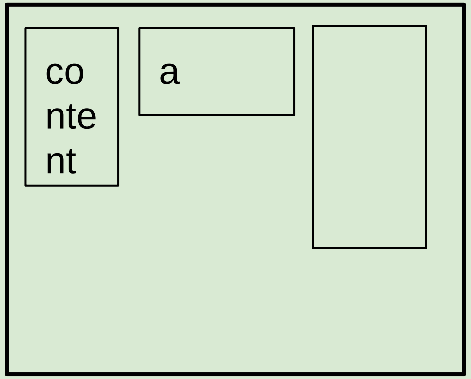
1. Starts on same line
2. Takes up only width required for content
3. Cannot set width/height

default for: span, img, a, etc.



inline-block

1. Like inline
2. Can set width/height



DISPLAY: BLOCK

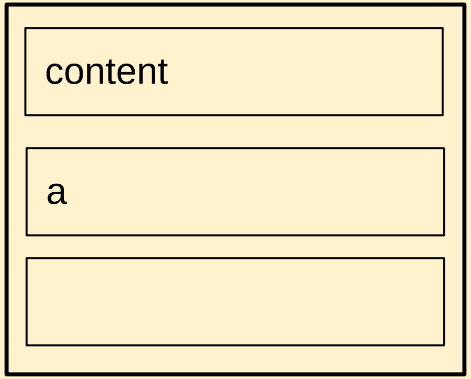
A **block-level element** always starts on a **new line** and takes up the **full width** available—meaning it stretches out to the left and right as far as it can.

default for: div, h1, p, form, header, footer, section, etc.

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

block

1. Starts on a newline
2. Takes up full available width
3. can set width/ height



DISPLAY: INLINE

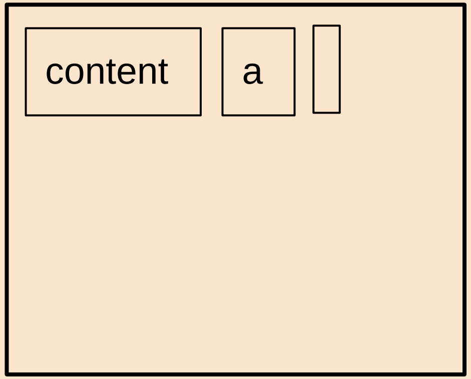
An **inline element** does not start on a new line and only takes up **as much width as necessary**.

default for: span, em, etc.

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

inline

1. Starts on same line
2. Takes up only width required for content
3. It accepts margin and padding while ignoring height and width



ELEVATE YOURSELF

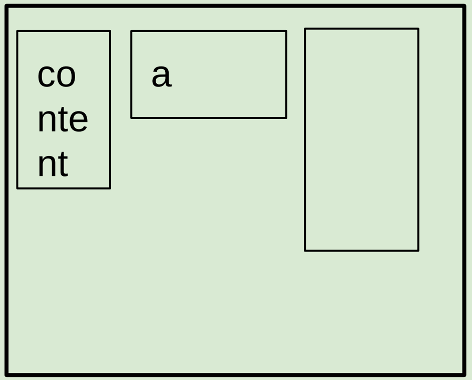
DISPLAY: INLINE-BLOCK

Similar to **inline**, but allows you to set a **width** and **height**

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

inline-block

1. Like inline
2. Can set width/height



DISPLAY: NONE

```
{
                display: none;
}
```

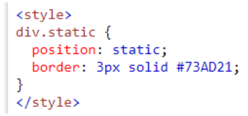
none

1. hides the element from view

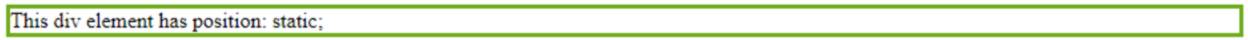
ELEVATE YOURSELF

HTML Elements: Positioning

- All elements have a default flow, a position they will fall into in the abcense of additional instructions. This is known as “static” flow. There are additional defined positions:
  - **static**: “default” value. Put an element in the page normally

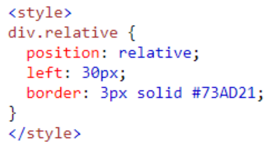


```
<style>
div.static {
  position: static;
  border: 3px solid #73AD21;
}
</style>
```

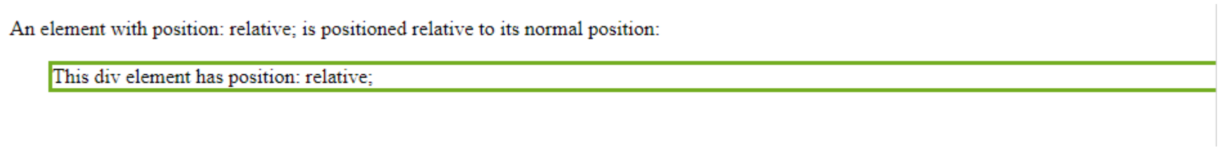


This div element has position: static;

  - **relative**: “relative” to what it would be positioned per the normal flow. (Hard to explain, we’ll do an example)



```
<style>
div.relative {
  position: relative;
  left: 30px;
  border: 3px solid #73AD21;
}
</style>
```

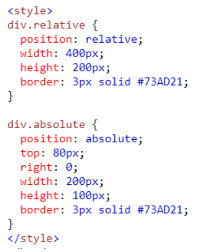


An element with position: relative; is positioned relative to its normal position:

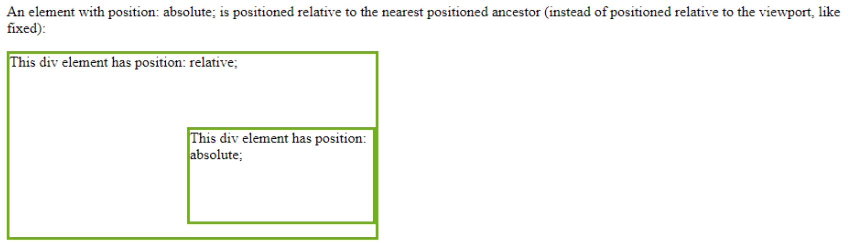
This div element has position: relative;

HTML Elements: Positioning

- All elements have a default flow, a position they will fall into in the abcense of additional instructions. This is known as “static” flow. There are additional defined positions:
  - **absolute**: positioned relative to its ancestor.



```
<style>
div.relative {
  position: relative;
  width: 400px;
  height: 200px;
  border: 3px solid #73AD21;
}

div.absolute {
  position: absolute;
  top: 80px;
  right: 0;
  width: 200px;
  height: 100px;
  border: 3px solid #73AD21;
}
</style>
```

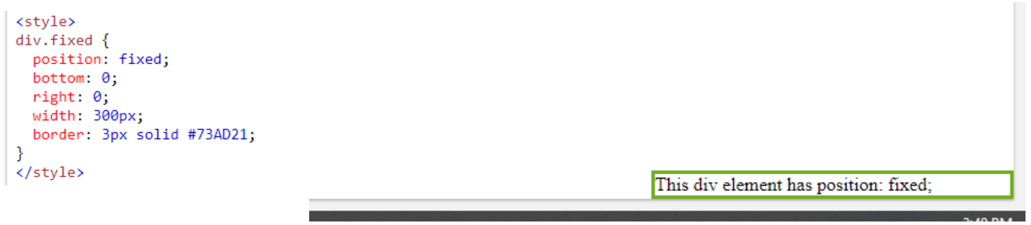


An element with position: absolute; is positioned relative to the nearest positioned ancestor (instead of positioned relative to the viewport, like fixed):

This div element has position: relative;

This div element has position: absolute;

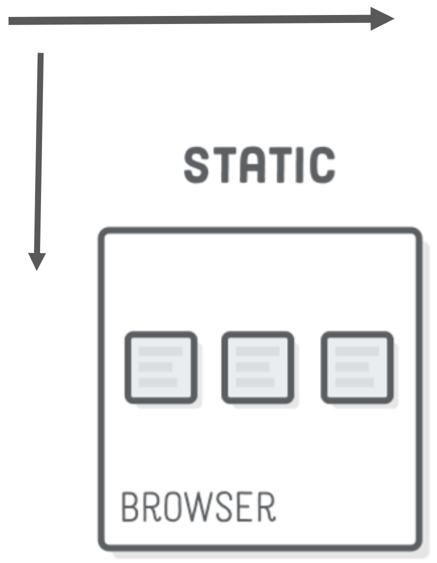
  - **fixed**: positioned relative to your screen, will follow you as you scroll.



```
<style>
div.fixed {
  position: fixed;
  bottom: 0;
  right: 0;
  width: 300px;
  border: 3px solid #73AD21;
}
</style>
```

This div element has position: fixed;

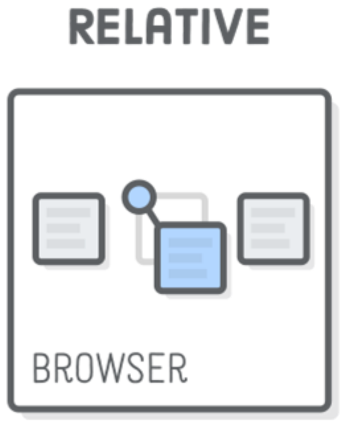
POSITION STATIC



- The normal flow of a page is for elements to appear left to right and top to bottom based on the order in which they appear in the HTML document and the rules of block and inline display.
- Static position by default means the element conforms to normal flow.

Positioning Demonstration

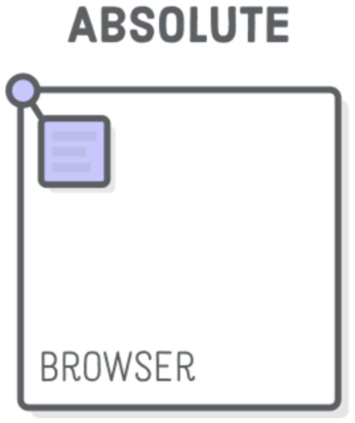
POSITION RELATIVE



- Relative position means relative to where it would otherwise be positioned in the normal flow.
- You can set the top, right, bottom, and left positioning attributes.

Positioning Demonstration

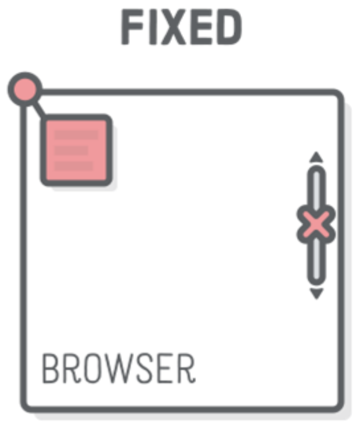
POSITION ABSOLUTE



- Absolute position places the element relative to the parent ancestor—that is, the containing element—exactly where you specify. If its it parent is the document body, it will move along with page scrolling.
- These elements are removed from the flow of the page.
- You can set the top, right, bottom, and left positioning attributes

Positioning Demonstration

FIXED POSITION



- Fixed position is relative to the browser window and does not scroll with the page.
- You can set the top, right, bottom, and left positioning attributes, Not setting a location causes the element to be fixed to wherever it normally shows up on the page.

Positioning Demonstration

Full list of Units of Measurement

Units of Measurement

Absolute Units - fixed size

px	pixels	1px = 1/96 of 1 inch. However, pixels are relative to the device, as 1px is 1 dot on the viewing device. Viewing devices have different pixel sizes.

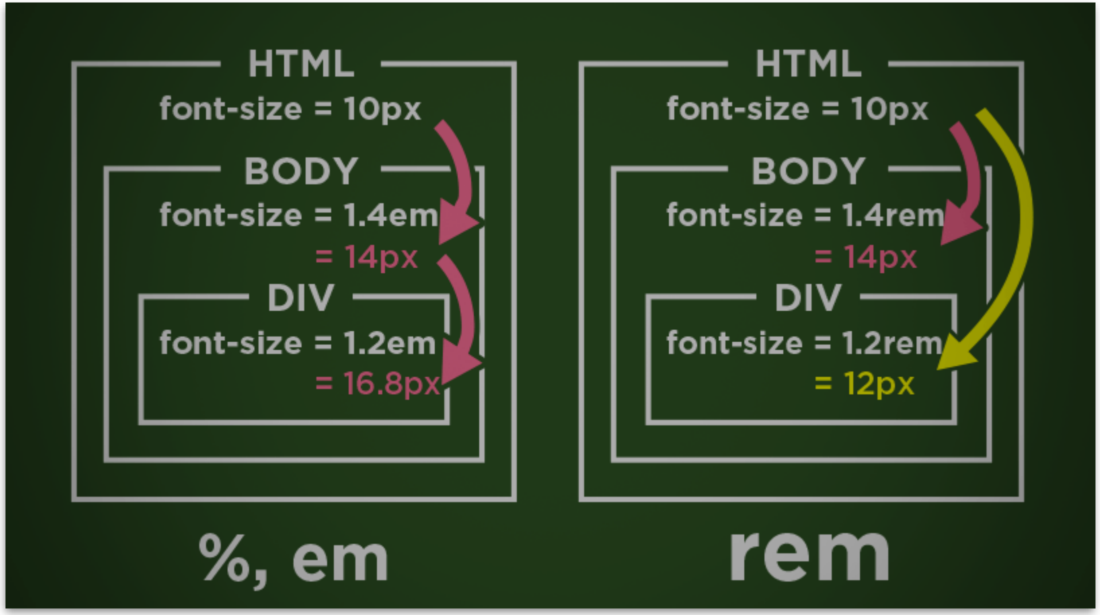
Relative Units - size is relative to another property

em	Relative to the font-size of the element. (2em = 2xs the current font size)
rem	Relative to the font-size of the root element.
vw	Relative to 1% of the width of the viewport
vh	Relative to 1% of the height of the viewport
%	Relative to the size of the parent element

Absolute Units of measurement should be avoided, and do not scale with different devices. Though they are sometimes still needed.

Relative Units of measurement should be used when possible, as they scale with different devices.

EM vs REM


HTML
font-size = 10px
BODY
font-size = 1.4em
= 14px
DIV
font-size = 1.2em
= 16.8px
%, em
HTML
font-size = 10px
BODY
font-size = 1.4rem
= 14px
DIV
font-size = 1.2rem
= 12px
rem