CSS
Layout

Naming Elements with an ID or Class

To have more control over your css you can name your HTML elements with an ID or Class. Identifiers (ID) are unique names given to an element. A class can be used to name more than one element.

Name your `<div>` with an ID or class within the opening tag of the element:

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

<div class="left-column"> </div>
```

In your CSS document, ID selectors begin with a hashtag, and class selectors begin with a period:

ID

```
html  <div id="sidebar">  </div>
css   #sidebar { float: left; }
```

Class

```
html  <div class="left-column">  </div>
css   .left-column { float: left; }
```

Display

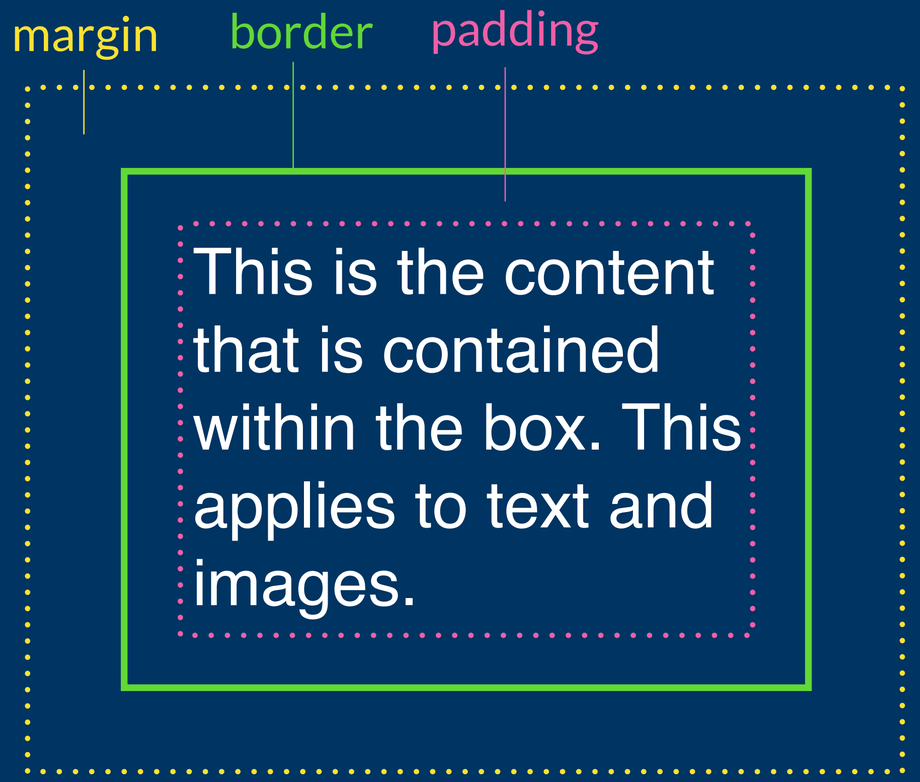
The display property allows you to turn a block element to an inline element and vice versa.

display:inline
causes a block-level element to act like an inline element (i.e. making list items appear like a horizonal nav)

display:block
causes an inline element to act like a block-level element

Box Model

When setting heights and widths for an element in CSS you must take into account accurate measurements for all your boxes' properties (i.e. padding, margins, etc).




```
.column {
    width: 300px;
    padding: 10px;
    margin: 20px;
    border: 5px solid green;
}
```

Total Width of Box= 370px

box-sizing: border-box

The above CSS declaration is fairly new, but a bit of a game changer. It takes away the need to account for and add the padding and border widths into the total width of the box.

The box-sizing property allows us to include the padding and border in an element's total width and height.

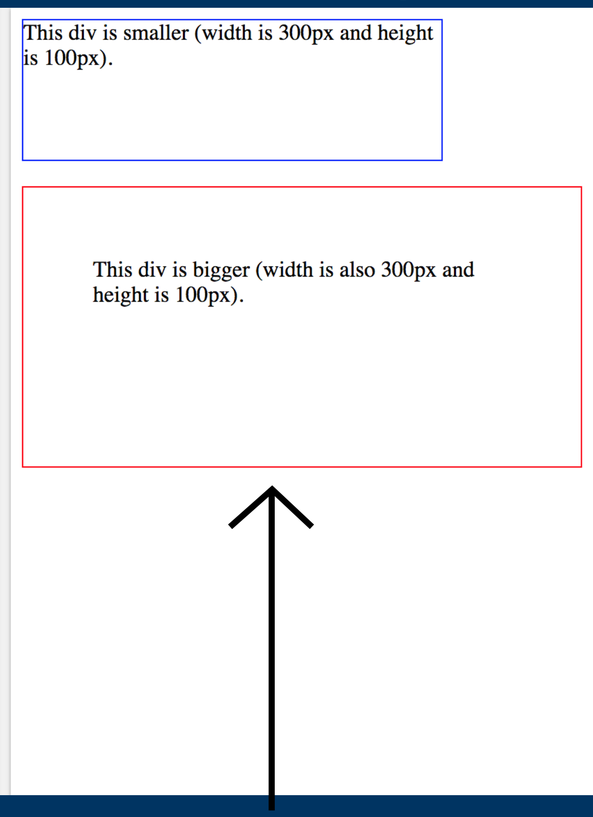
It is good practice for most developers to now include this universally in their CSS.

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

Without box-sizing

300px width + 1px border = 302px total

```
<!DOCTYPE html>
<html>
<head>
<style>
.div1 {
  width: 300px;
  height: 100px;
  border: 1px solid blue;
}

.div2 {
  width: 300px;
  height: 100px;
  padding: 50px;
  border: 1px solid red;
}
</style>
</head>
<body>

<div class="div1">This div is smaller (width
is 300px and height is 100px).</div>
<br>
<div class="div2">This div is bigger (width
is also 300px and height is 100px).</div>

</body>
</html>
```



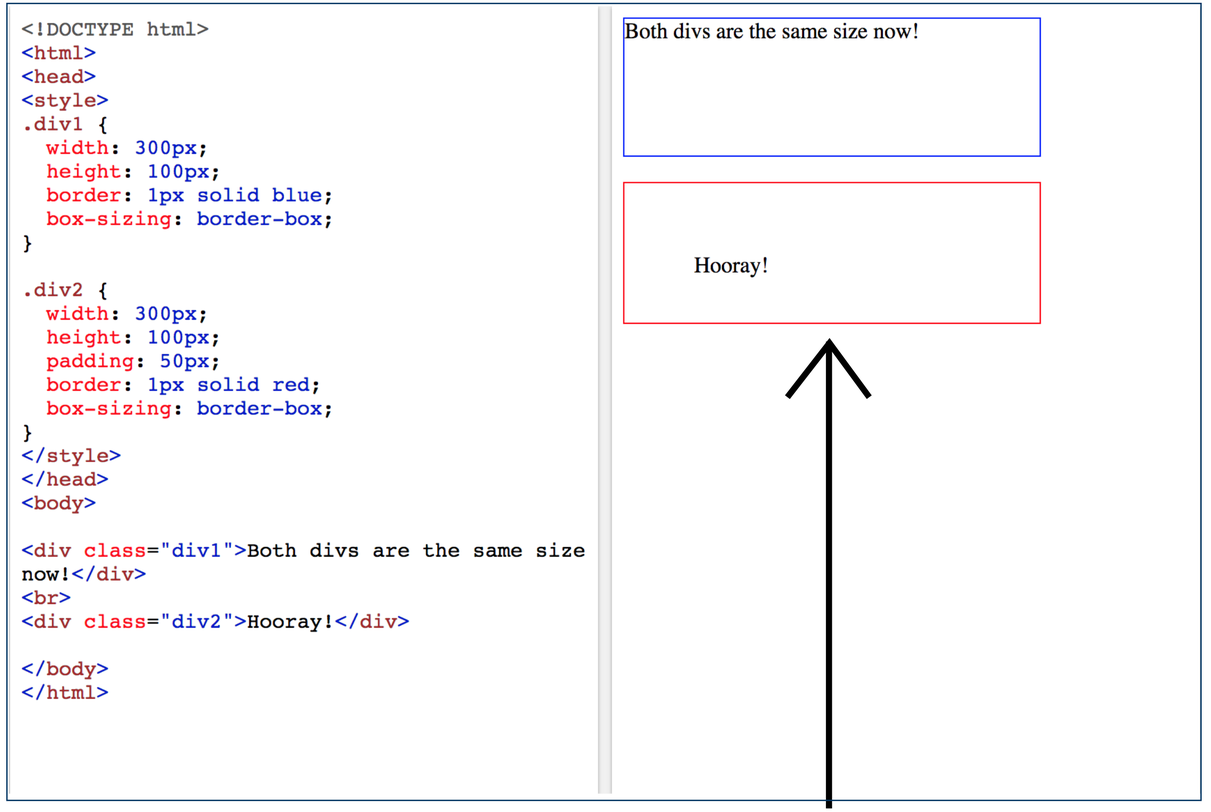
300px width + 50px padding + 1px border = 402px total width

Math

With box-sizing

300px width

```
<!DOCTYPE html>
<html>
<head>
<style>
.div1 {
  width: 300px;
  height: 100px;
  border: 1px solid blue;
  box-sizing: border-box;
}

.div2 {
  width: 300px;
  height: 100px;
  padding: 50px;
  border: 1px solid red;
  box-sizing: border-box;
}
</style>
</head>
<body>

<div class="div1">Both divs are the same size
now!</div>
<br>
<div class="div2">Hooray!</div>

</body>
</html>
```



300px width

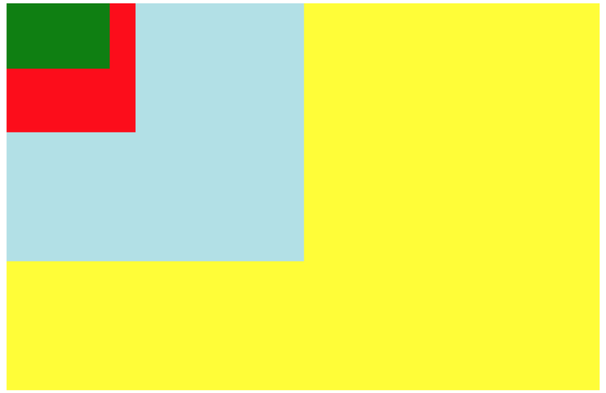
No Math

Box Dimensions

When using **percentages**, the size of the box is relative to the size of the browser window, or if the box is encased within another box, it is a percentage of the size of the containing box.

When using **ems**, the size of the box is based on the the size of the text that it sits inside of.

```
<!DOCTYPE html>
<html>
<head>
<style>

#yellowbox {
height: 300px;
width: 50%;
background-color: yellow;
}

#bluebox {
height: 200px;
width: 50%;
background-color: powderblue;
}

#redbox {
font-size: 20px;
height: 100px;
width: 5em;
background-color: red;
}

#greenbox {
height: 50px;
width: 5rem;
background-color: green;
}

</style>
</head>
<body>

<div id="yellowbox">
<div id="bluebox">
    <div id="redbox">
        <div id="greenbox">
        </div>
    </div>
</div>
</div>

</body>
</html>
```



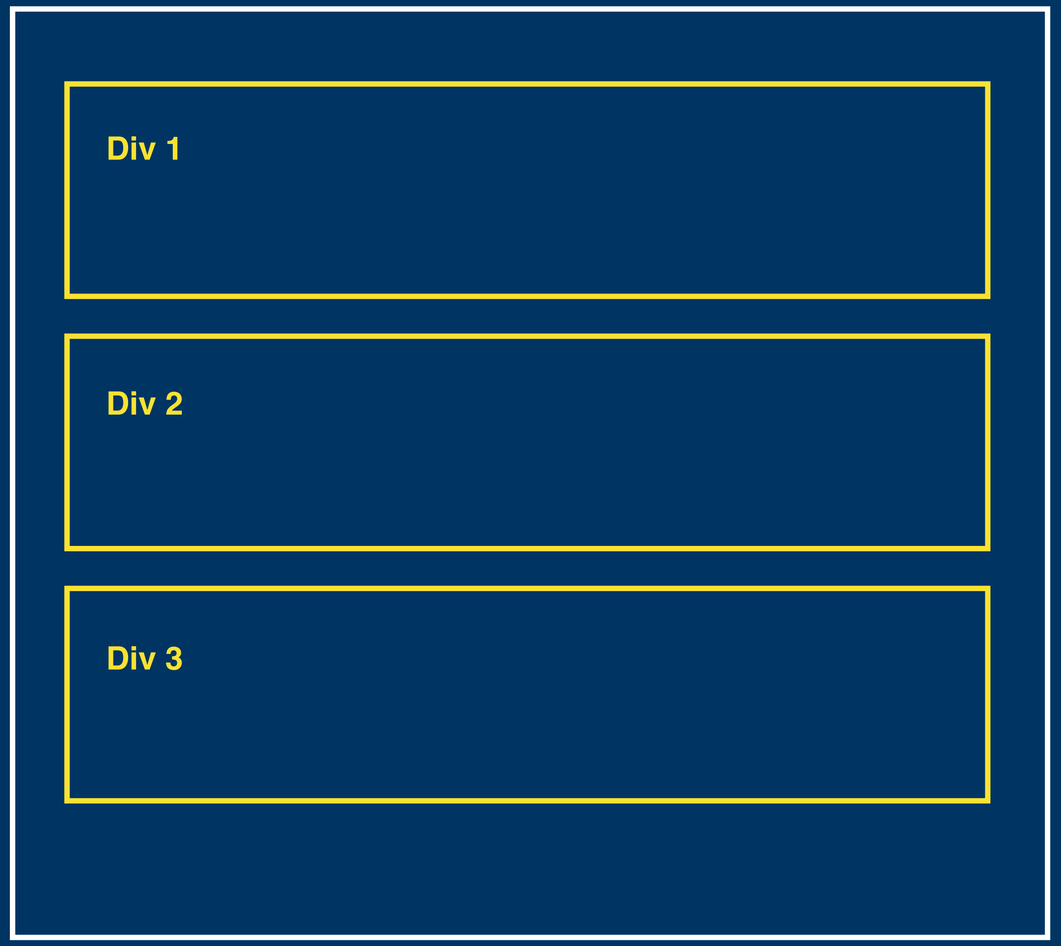
Layouts

Fixed layout utilizes pixel measurements that do not change sizes as the user increases or decreases the size of their browser window.

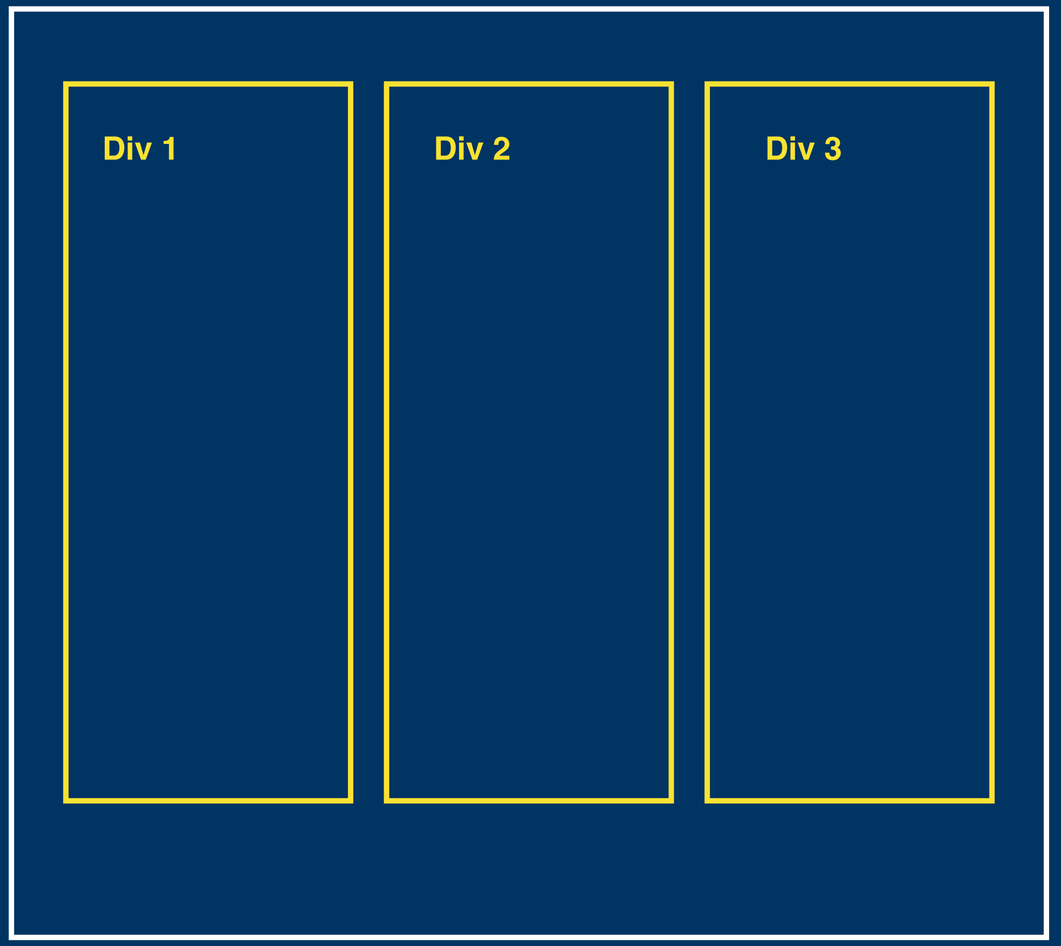
Fluid layout utilizes percentage measurements and stretch and contract as the user increases or decreases the size of their browser window.

Floating Elements

Floating an element allows you to take that element out of the normal flow and position it to the far right or left of it’s parent box. This is how you create multi-column layouts.



Normal flow



float: left;

Positioning Elements

position: fixed

The element will not remain in the natural flow of the page. It will position itself according to the viewport.

It will respond to the following properties:

top
bottom
left
right
z-index

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

<h2>position: fixed;</h2>

<p>An element with position: fixed; is positioned
relative to the viewport, which means it always stays in
the same place even if the page is scrolled:</p>

<div class="fixed">
This div element has position: fixed;
</div>

</body>
</html>
```

position: fixed;

An element with position: fixed; is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled:

This div element has position: fixed;

position: absolute

The element will not remain in the natural flow of the page. It will position itself according to its parent container—which must be set to **position: relative**

It will respond to the following properties:

top
bottom
left
right
z-index

```html
<!DOCTYPE html>
<html>
<head>
<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>
</head>
<body>

<h2>position: absolute;</h2>

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

<div class="relative">This div element has position:
relative;
  <div class="absolute">This div element has position:
absolute;</div>
</div>

</body>
</html>
```

position: absolute;

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;

z-index

z-index allows you to “bring to front” or “bring to back” an element that is positioned **absolute** or **fixed**, allowing you to control which box appears on top.

The value of the z-index property is a number, and the higher the number the further that element is on top.

```
position: absolute;
z-index : 2;
```

```html
<!DOCTYPE html>
<html>
<head>
<style>
img {
  position: absolute;
  left: 10px;
  top: 10px;
  z-index: -1;
}
</style>
</head>
<body>

<h1>The z-index Property</h1>

<img src="w3css.gif" width="100" height="140">

<p>Because the image has a z-index of -1, it will be placed behind the heading.</p>

</body>
</html>
```

The z-index Property

Because the image has a z-index of -1, it will be placed behind the heading.