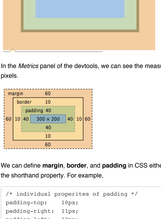
Lesson 4 - Notes

Nathan Stilwell March 4, 2013

The Box Model

When HTML is rendered in the browser, all elements are rendered as "a box" with 4 basic parts; Margin, Border, Padding, and Content.

If we inspect an element, the Chrome devtools will illustrate these components for us.



In the *Metrics* panel of the devtools, we can see the measurements of each part of the box model in pixels.

We can define **margin**, **border**, and **padding** in CSS either using individual properties or by using the shorthand property. For example,

```
/* individual properites of padding */
padding-top:    10px;
padding-right:  11px;
padding-left:   12px;
padding-bottom: 13px;

/* shorthand properties of padding */
padding : 10px 11px 12px 13px;
```

When using a shorthand property we can define the value using space seperated parameters.

Shorthand properties for **margin** and **padding** can take 1 to 4 parameters.

```
/* top, right, bottom, left */
padding : 1px 13px 17px 7px;

/* top, left/right, bottom */
margin : 5px 10px 5px;

/* top/bottom, left/right
padding : 10px 20px;

/* top/bottom/left/right */
margin : 15px;
```

See below for the shorthand for border

Margin

The area outside of the border, fills the space of the parent on block level elements. Can be specified in px, em, or percentage.

If the defined amount of margin does not fill the parent, the remaining margin will be extended. For example,

.container { width: 520px; }

.content { margin: 25px 10px 15px 6px; width: 200px; }

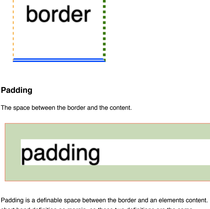
In the above example, we are defining a right margin of 6px. But we can see that the right margin is much more than 6px. This is because the right margin is filling the difference between the parent container's width of 520px and our element's width of 200px. To calculate the width of the right margin we can take the container width of 520px, subtract the elements width of 200, then subtract the left margin of 25px.

```
margin-right = 520px - 200px - 25px
margin-right = 295px
```

Since we have defined the right margin as 10px, the Chrome devtools with tell us that it is 10px.

Using this auto filling mechanism, we can ask the browser to automatically calculate the space on both sides of our element centering it inside the parent.

Centering an Element



Here we have a container that is 440px wide and inside it we have an element that is 100px wide. Using this difference we set the left and right margins to `auto` and the browser will calculate the difference in width and divide it evenly to center the element.

```
<!-- html -->
<div class="box-example margin-example">
    <div class="margin-auto">
        Centered
    </div>
</div>

/* css */
.box-example {
    border: 3px dotted;
    width: 440px;
}
.margin-auto {
    margin: 20px auto;
    width: 100px;
}
```

In the devtools we can see that the left and right margins were divided as evenly as possible.

Border

The `border` property is shorthand for three other shorthand properties; `border-width`, `border-style`, `border-color`.

```
/*     width style color */
border : 1px solid black;
```


```
border : 4px dotted tomato;
```

`border-width`, `border-style`, `border-color` are themselves shorthand properties for each individual side of the box, like margin and padding.

```
border-width: 4px 6px 8px 2px;
border-style: solid dotted double dashed;
border-color: red green blue orange;
```

would look like this

Padding

The space between the border and the content.

Padding is a definable space between the border and an elements content. Padding uses the same short hand definition as margin, so these two definitions are the same.

```
padding : 35px;

/* or */

padding-top: 35px;
padding-right: 35px;
padding-bottom: 35px;
padding-left: 35px;
```

Content

Content the area at the core of the box model. Using CSS we can define properties for the content's height and width. Those CSS properties are:

- `height`
- `min-height`
- `max-height`
- `width`
- `min-width`
- `max-width`

Height and width can be defined in px, em, or %. Using a combination can create a responsive element that is based on the windows width, with a minimum and a maximum. For example:

```
.example {
    width: 70%;
    min-width: 400px;
    max-width: 700px;
}
```

This example will be 70% of the window, as long as it's at least 400px width and at most 700px.

Float

Normal Page Flow

In **normal page flow** elements are arranged based on their *display type*. Most tags fall into two default display types, **block** and **inline**.

Block Elements

Block elements will *automatically* inherit the width of their parent. Since they will take the entire width of their parent, they force elements after them to render underneath them causeing what looks like a "line break". For example,

```
<article style="width: 500px;">
    <div>div</div>
</article>
```

`article` and `div` are both block elements. The `article` is given a width of 500px. The `div` isn't given a width so it will inherit it's parent's width; so it will also be 500px wide.

Block elements will automatically have a height equal to the height of their content.

Inline Elements

Inline elements will automatically have a width equal to the width of their content. Since they don't take up the entire width of their parent, other inline element will be horizontally aligned with them until their collective widths fill the width of the parent.

more to come ....