ITSE412 – The CSS Box Model

Cascading Style Sheets: Margin, Border, Padding, and Positioning

The CSS Box Model

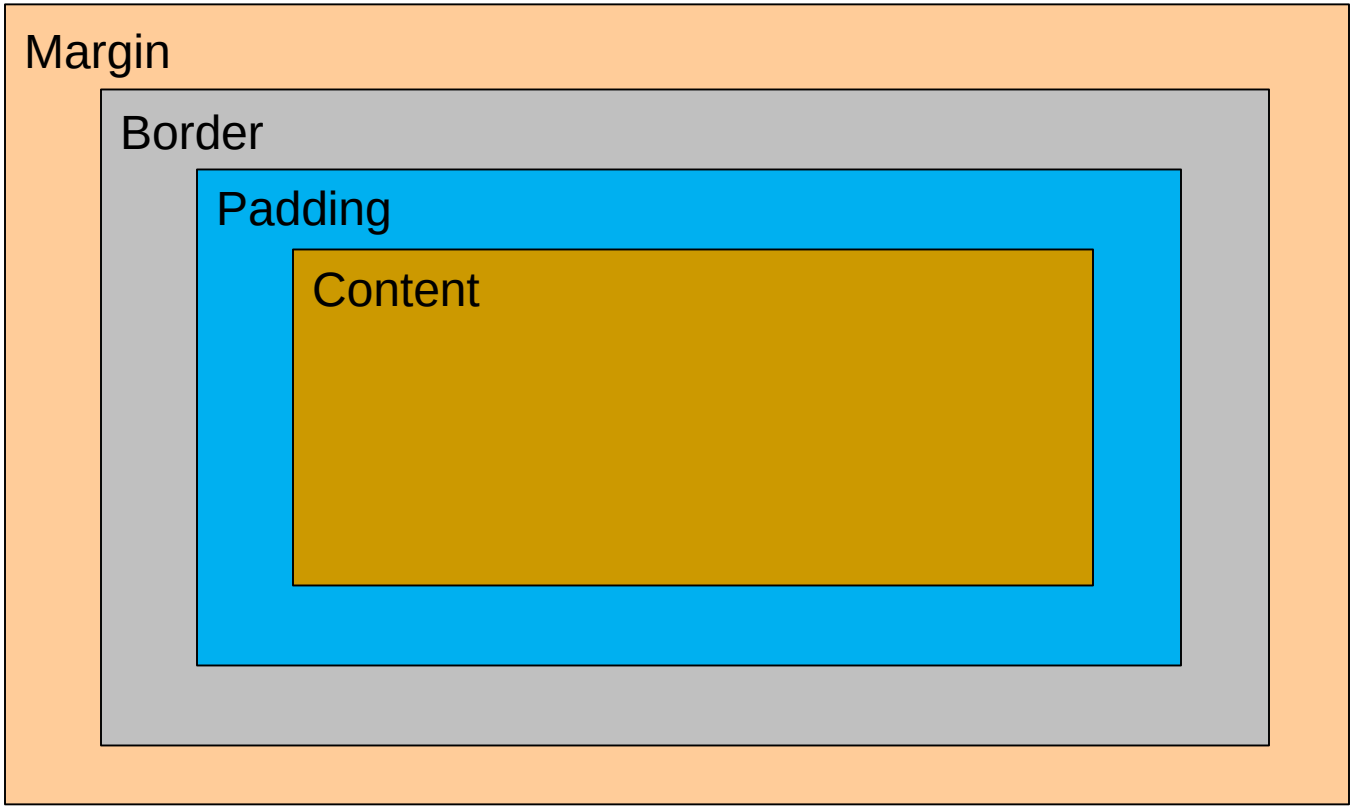
The box model is a tool we use to understand how our content will be displayed on a web page.

- Each HTML element takes up a "box" or "container" of space on the webpage.
- Each box size is affected by its margins, borders, and padding.
- By knowing how to calculate the dimensions of each box, we can accurately predict how elements will lay out on the screen.

The understanding of the box model concept is very crucial for web designers. It would be difficult and frustrating to create a website without understanding this concept.

Box Components

Each element on the page has the following components:



The easiest way to understand these components is to use the element.

Introducing the Element

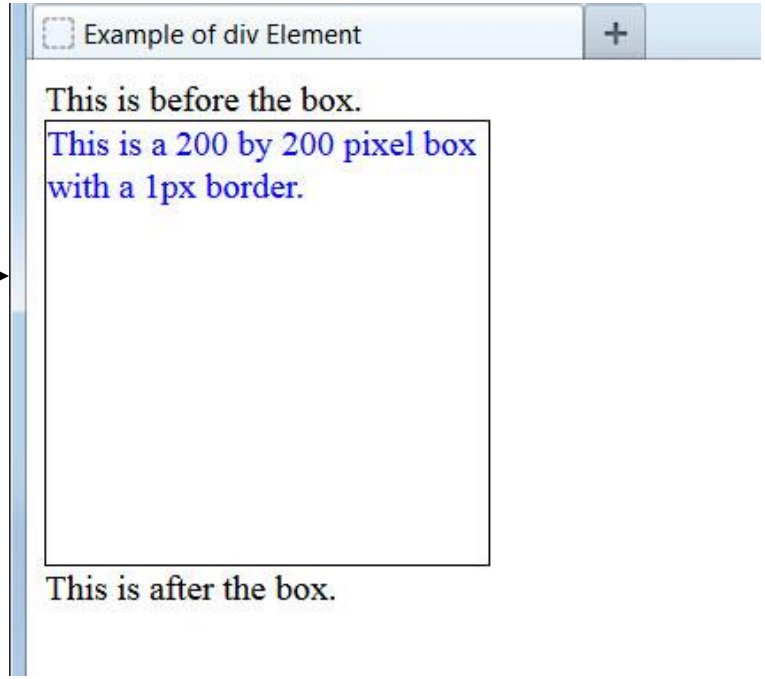
The ("division") element groups other elements on the screen.

- By setting the **width** and **height** attributes via CSS, we can reserve a precise amount of space on our page for specific content.
- The actual content is contained within the opening and closing tags.
- When we apply CSS styling directly to the element, all the elements contained within that will inherit that style.
- By using multiple elements as building blocks, we can design an entire web page layout.

Example Element

Let's create a box on the screen and add a border around it:

```
<style type="text/css">
  .box200 {
    width: 200px;
    height: 200px;
    border: 1px solid black;
    color: blue;
  }
</style>
...
This is before the box.
  <div class="box200">
This is a 200 by 200 pixel box with
    a 1px border.
  </div>
This is after the box.
...
```

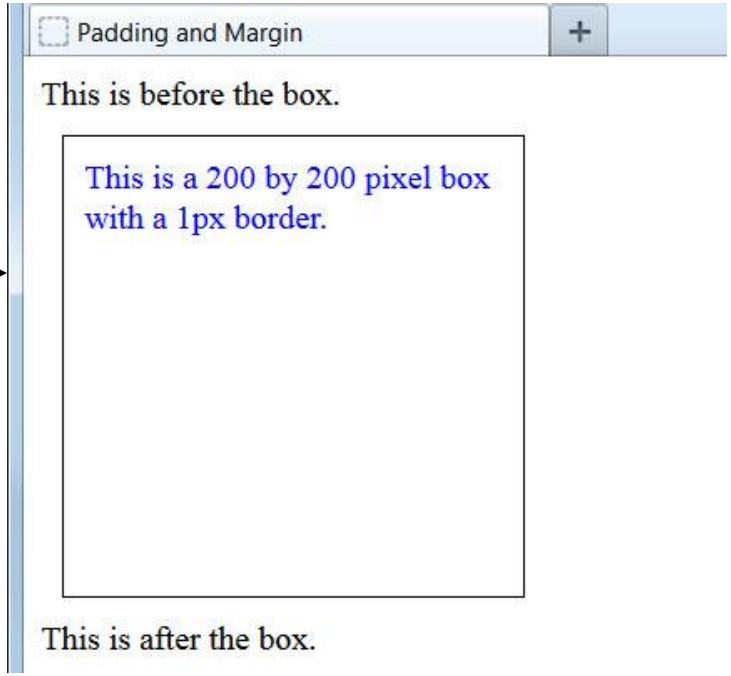


Notice that there is almost no space separating the text from the box border. HTML Elements such as paragraphs, headers, and lists automatically insert padding and margins, but plain text does not do so.

Adding Padding and Margin

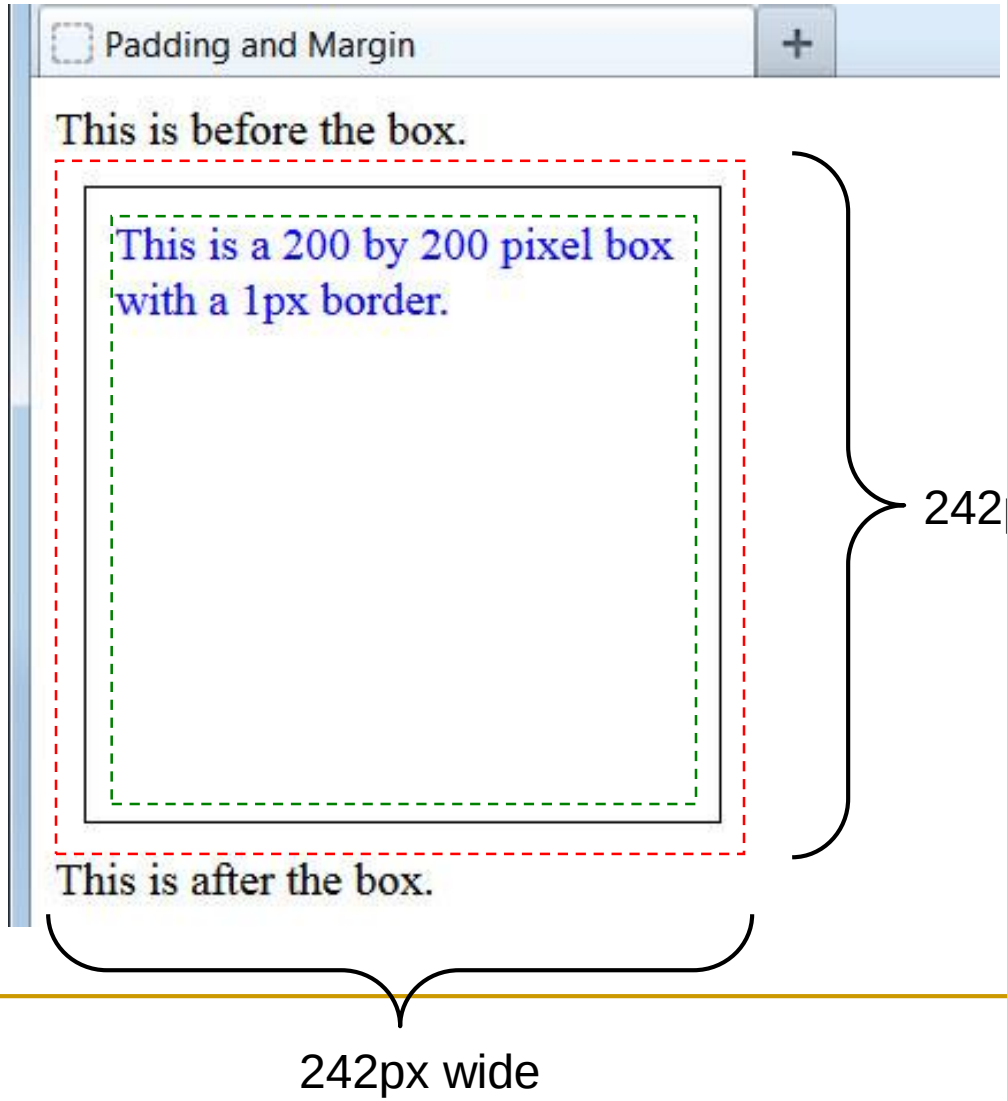
Let's create some space between elements by adding both padding and margin:

```
<style type="text/css">
   .box200 {
     width: 200px;
     height: 200px;
     border: 1px solid black;
     color: blue;
     padding: 10px;
     margin: 10px;
   }
</style>
...
```



The 10 pixel padding adds buffer space, on all four sides, between the content and border. The 10 pixel margin adds buffer space, on all four sides, between the border and surrounding elements.

Padding and Margin Illustrated



The dotted red line shows the margin's outer boundary and the dotted green line shows the padding's inner boundary.

When we define the width and height of a element, these dimensions apply only to the actual content, not to the padding, border, or margin.

Calculating Total Dimensions

When we are planning our page, we have to calculate exactly how much screen space a or any other element will use. The formula is:

- Total element width = defined width + left padding + right padding + left border + right border + left margin + right margin.
- Total element height = defined height + top padding + bottom padding + top border + bottom border + top margin + bottom margin.

In our previous example:

- Total width = 200px + 10px + 10px + 1px + 1px + 10px + 10px = 242px.
- Total height = 200px + 10px + 10px + 1px + 1px + 10px + 10px = 242px.

Padding, borders, and margins do not have to be the same on all four sides, as they are in this example. Let's see how to set these individually.

Setting Individual Margins

We can set the specific margin sizes by using these four properties:

margin-top:
margin-right:
margin-bottom:
margin-left:

In practice, web designers use a **shorthand** method that condenses the declaration to a single line as follows:

margin: 10px; → Sets all four margins to be 10px.

margin: 10px 5px; → Sets the top and bottom margins as 10px and the left and right margins as 5px.

margin: 20px 10px 5px 15px; → Sets the top margin as 20px, the right margin as 10px, the bottom margin as 5px, and the left margin as 15px.

Setting Individual Padding

We can set specific padding sizes by using these four properties:

padding-top:
padding-right:
padding-bottom:
padding-left:

Again, these are rarely used. Instead, the same shorthand method is employed:

padding: 5px; → Sets padding as 5px on all four sides.

padding: 25px 5px; → Sets top and bottom padding as 25px and left and right padding as 5px.

padding: 50px 10px 15px 5px; → Sets top padding as 50px, right padding as 10px, bottom padding as 15px, and the left padding as 5px.

Setting Individual Borders

Customizing borders is a bit complicated. Since they are visible on the page, we have to specify style, thickness, and color. Accordingly, there are more properties available:

border-top-width:	border-top-style:	border-top-color:
border-right-width:	border-right-style:	border-right-color:
border-bottom-width:	border-bottom-style:	border-bottom-color:
border-left-width:	border-left-style:	border-left-color:

To make things easier, we can use the shorthand method as before:

```
border-width: 10px;
border-style: solid dashed;
border-color: blue red orange gray;
```

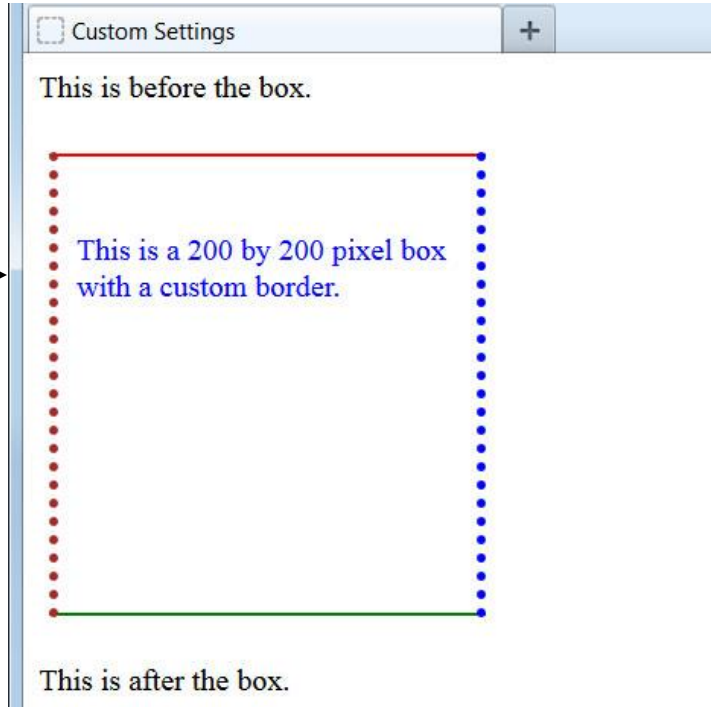
If the three border properties will be identical on all four sides, we can use a single-line border shorthand, as we did in an earlier lesson:

```
border: 5px solid blue;
```

Example of Customized Settings

Here we have set custom padding, borders, and margins:

```
<style type="text/css">
  .box200 {
    width: 200px;
    height: 200px;
    color: blue;
    padding: 40px 10px 0px 10px;
    margin: 25px 5px;
    border-width: 2px 5px;
    border-style: solid dotted;
    border-color: red blue green maroon;
  }
</style>
```



A helpful way to remember the order of shorthand settings is that it starts at noon and goes clockwise around: top, right, bottom, left.

Introduction to CSS-Positioning

- CSS-P allows the web developer to position HTML-content anywhere on the web page.
- CSS-P should normally be applied to the **DIV** tag only- any other HTML elements can be placed within the DIV tag .
- The DIV tag was originally created to mark a section division within a larger HTML page, but now it is used to position HTML elements.
- CSS-P is replacing tables as the primary means for page layout in HTML.

CSS-P Basic Positioning

- The positioned DIV tag should have ONLY CSS-P properties assigned to it.
- Text-formatting properties should be assigned to the <p> tag, the heading tags, etc, as usual.
- You must use CSS-P very carefully, and only within the first 300-400 pixels in height, and 500-700 pixels in width.
- You should also use CSS-P by itself, not in combination with regular HTML styling whenever possible.

CSS-P example 1

```
<html> <head> <title>Example CSS-P Page</title>
    <style type="text/css">
        #box1 { position:absolute; left:100px; top:30px; }
    </style>

</head>
<body>
Text before the image.
<div id= "box1">
        <img src="flowers.jpg" width="500" height="100">
</div>
Text after the image.

</body>
</html>
```

- In the above example, Box1 upper-left-hand corner is positioned 100 pixels from the left, and 30 pixels from the top.

CSS-P example 1

- Again, there are a minimum of THREE essential CSS-P properties required to place an element: position, left, and top.
  - the position property declares the type of positioning (relative, fixed, absolute, ..);
  - the left property gives the x-coordinate (in pixels) for the positioned element;
  - the top property gives the y-coordinate (in pixels) for the positioned element.

CSS-P Required Properties

- **Property: position**
  - values: relative, fixed , absolute
- Example: *position: absolute;*
- Relative positioning is hard to control, and rarely used;
- Fixed positioning removes an element from the flow of HTML code and "fix" it to the web browser. When elements on an HTML page scroll up or down the "fixed" element does not move.
- Absolute positioning places the element in relation to the upper-left-hand corner of the web page. The absolutely positioning is the most commonly used type of positioning.

CSS-P Required Properties

- **Property: left**
  - Values: 10px, 172px, 59px, etc
- **Example:** left:10px;
- **Description:** sets the x-coordinate for the positioned element, in pixels.

CSS-P Required Properties

- **Property: top**
  - Values: 10px, 172px, 59px, etc
- **Example: top:25px;**
- **Description:** sets the y-coordinate for positioned element, in pixels.

CSS-P example 2

```
<html> <head>
<title>Example CSS-P Page</title>
 <style type="text/css">
    #box2 { position: absolute; left:100px; top:30px; }
 </style>
</head>
<body>
<div id="box2" >
      <table width="200“ border="1">
      <tr>
           <td>1</td>
           <td>2</td>
      </tr>
      <tr>
           <td>3</td>
           <td>4</td>
      </tr>
       </table>
</div>
</body>
</html>
```

Positioning Text: The width Property

- **You can position paragraphs of text in the same way that you position images, using a positioned DIV tag.**

```
<html> <head>
<title>Example CSS-P Page</title>
<style type="text/css">

#box3 { position: absolute; left:100px; top:30px; }
  </style>
</head>
<body>
<div id="box3">
    <p border="2" > Here is some text. It is not very exciting text, but it is long.
    Long, long text. It goes on and on and on. Why, look at all this text! I wonder if
    it will ever say anything interesting? No, I think not! But it sure is long!</p>
</div>
<p>Here is another paragraph of text. This one is short.</p>

</body>
</html>
```

Positioning Text: The width Property

- **Property: width**
  - **Values: any integer-based pixel value (100px, 250px, etc)**
- **Description: The width property sets the width of a positioned element.**
- **Example: width:250px;**

Positioning Text: The width Property

```
<html> <head>
<title>Example CSS-P Page</title>
<style type="text/css">
<!–
#box4{ position:absolute; left:100px; top:30px; width:250px; }
//-->
</style>
</head>
<body> <div id="box4">
<p>Here is some text. It is not very exciting text, but it is long. Long, long text. It goes on and on and on. Why, look at all this text! I wonder if it will ever say anything interesting? No, I think not! But it sure is long!</p>

<p>Here is another paragraph of text. This one is short.</p>
</div>
</body>
</html>
```

Making a Colored DIV Box

- You can turn a positioned DIV tag into a colored box. This feature is particularly useful when creating application "screens" for DHTML purposes.
- There is only one real way to create these colored boxes which are supported by ALL browsers properly. First, you must create a DIV tag on your page and position it. In this case, however, the DIV tag will be empty.

Making a Colored DIV Box

```
<html> <head>
<title>Example CSS-P Page</title>
<style type="text/css">
        #box4{ position:absolute; left:100px; top:30px; }
</style>
</head>
<body>
        <div id=“box4">   </div>
</body>
</html>
```

- This isn't complete. To create the box, you must define BOTH the **width** and **height** properties in your CSS definition using pixel values.

Making a Colored DIV Box

- **Property: width**
  - **Values: 100px, 250px, etc…**
- **Description:** defines the width of a positioned element.
- **Property: height**
  - **Values: 100px, 250px, etc>>>**
- **Description:** defines the height of a positioned element.
- **Property**: background-color
  - Values: any hex code color value (#CC66CC, #FFCCCC, etc)
- **Description**: defines background color for a positioned
- **Property**: border
  - Values: border type (solid), width of border (1px), color
- Example: border: solid 1px #CC99CC;

Making a Colored DIV Box

```
<html> <head>
<title>Example CSS-P Page</title>
<style type="text/css">
#box4 {
        position:absolute;
        left:100px;
        top:30px;
        width:100px;
        height:100px;
        background-color:#996699;
        border:solid 1px #996699; }
        </style>
</head>
<body>
<div id="box4">   </div>
</body>
</html>
```