

Comp 125 - Visual Information Processing

Spring Semester 2018 - week 9 - friday

Dr Nick Hayward

CSS Basics - box model - part 4

Border

- **border area** extends **padding area** to area containing the borders
- it becomes the area inside the **border edge**
- define its dimensions as the width and height of the **border-box**
- calculated area depends upon the width of the border we set in the CSS
- set size of our border using the following properties in CSS,
 - *border-width*
 - *border*

Demo - CSS Box Model - Border

- [JSFiddle - CSS Box Model](#)

CSS Basics - box model - part 5

Margin

- **margin area** can extend this border area with an empty area
 - *useful to create a defined separation of one element from its neighbours*
- dimensions of area defined as width and height of the **margin-box**
- control size of our margin area using the following properties,
 - *margin-top, margin-right, margin-bottom, margin-left*
 - *margin* (sizes calculated clock-wise)

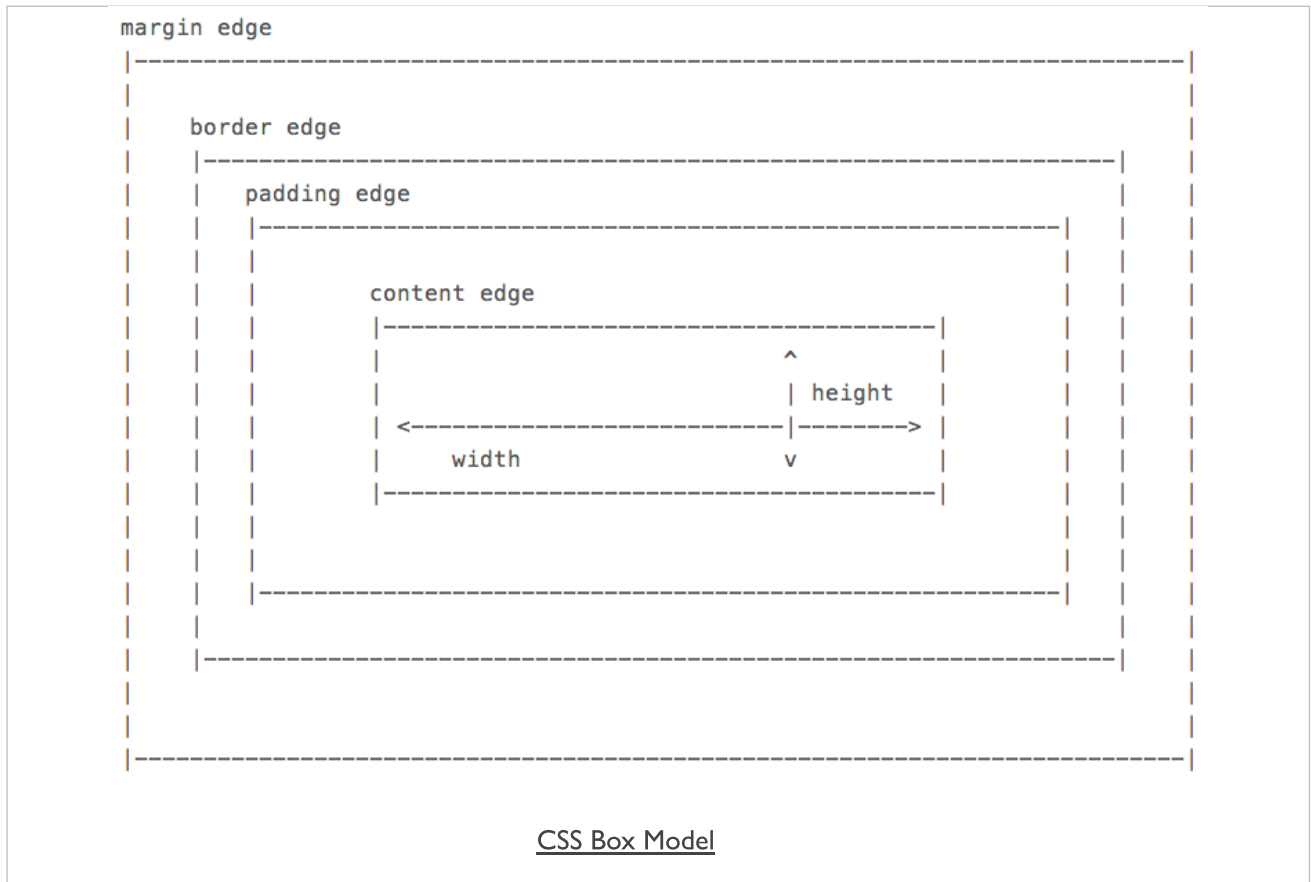
Demo - CSS Box Model - Margin

- JSFiddle - CSS Box Model

Demo - CSS Box Model

- Demo - CSS Box Model

Image - CSS Box Model



Source - MDN - CSS Box Model

CSS Basics - selectors

- **selectors** are a crucial part of working with CSS, JS...
- basic selectors such as

```
p {  
  color: #444;  
}
```

- above ruleset adds basic styling to our paragraphs
 - sets the text colour to HEX value 444
- simple and easy to apply
 - applies the same properties and values to all paragraphs
- specificity requires classes, pseudoclasses...

CSS Basics - classes

- add a **class** attribute to an element, such as a `<p>`
 - *can help us differentiate elements*
- also add a **class** to any DOM element
 - e.g. add different classes to multiple `<p>` elements

```
<p class="p1">paragraph one...</p>
<p class="p2">paragraph two...</p>
```

- we can now select our paragraphs by class name within the DOM
- then apply a **ruleset** for each class
- style this class for a specific element

```
p.p1 {
  color: #444;
}
```

- style all elements with the class p1, and not just `<p>` elements

```
.p1 {
  color: #444;
}
```

CSS Basics - pseudoclasses

- add a class to links or anchors, styling all links with the same ruleset
- we might also want to add specific styles for different link states
- styling links with a different colour
 - e.g. *whether a link has already been used or not*

```
a {  
  color: blue;  
}  
  
a:visited {  
  color: red;  
}
```

- visited is a CSS **pseudoclass** applied to the <a> element
- browser implicitly adds this pseudoclass for us, we add style

```
a:hover {  
  color: black;  
  text-decoration: underline;  
}
```

- pseudoclass for link element, <a>, hover

CSS Basics - complex selector - part I

- our DOM will often become more complicated and detailed
- depth and complexity will require more complicated selectors as well
- lists and their list items are a good example

```
<ul>
  <li>unordered first</li>
  <li>unordered second</li>
  <li>unordered third</li>
</ul>
<ol>
  <li>ordered first</li>
  <li>ordered second</li>
  <li>ordered third</li>
</ol>
```

- two lists, one unordered and the other ordered
- style each list, and the list items using rulesets

```
ul {
  border: 1px solid green;
}
ol {
  border: 1px solid blue;
}
```

Demo - Complex Selectors - Part I

- Demo - Complex Selectors Part I

CSS Basics - complex selector - part 2

- add a ruleset for the list items, ``
- applying the same style properties to both types of lists
- more specific to apply a ruleset to each list item for the different lists

```
ul li {  
  color: blue;  
}  
ol li {  
  color: red;  
}
```

- also be useful to set the background for specific list items in each list

```
li:first-child {  
  background: #bbb;  
}
```

- pseudoclass of `nth-child` to specify a style for the second, fourth etc child in the list

```
li:nth-child(2) {  
  background: #ddd;  
}
```

Demo - Complex Selectors - Part 2

- Demo - Complex Selectors Part 2

CSS Basics - complex selector - part 3

- style odd and even list items to create a useful alternating pattern

```
li:nth-child(odd) {  
  background: #bbb;  
}  
li:nth-child(even) {  
  background: #ddd;  
}
```

- select only certain list items, or rows in a table etc
 - e.g. every *fourth* list item, starting at the first one

```
li:nth-child(4n+1) {  
  background: green;  
}
```

- for **even** and **odd** children we're using the above with convenient shorthand
- other examples include
 - *last-child*
 - *nth-last-child()*
 - many others...

Demo - CSS Complex Selectors - Part 3

- Demo - Complex Selectors Part 3

References

- [MDN](#)
- [CSS documentation](#)
- [CSS Selectors](#)
- [W3Schools](#)
- [CSS](#)
- [CSS - Selectors Reference](#)