[Lesson 4]

Roi Yehoshua 2018

[DAN.IT EDUCATION]

What we learnt last time?

- Learn what is CSS and how it works
- CSS syntax
- Margins and Paddings, difference between them
- Basics of working with git

[Our targets for today]

- Browser default styles
- What is reset.css and why do we need it
- CSS block model
- box-sizing property
- Block display modes
- How to place several block elements on one line with “inline-block”
- How to create columns using and “inline-block”
- Elements tab in Chrome Developer Tools

[DAN.IT] EDUCATION

[Reset.css]

- Aside from styles set by user, browsers have their own default styles for many elements. They add by default margins and paddings.
- Reset.css can remove default margins and paddings and make styling more clear and obvious.
- There are many options of reset.css configuration - all of them may be easily found in the internet. One of them can be found here:

  https://meyerweb.com/eric/tools/css/reset/
- Reset.css can be added as external style via link tag:

```
<link rel="stylesheet" href="css/reset.css">
```

[DAN.IT] EDUCATION

[CSS Reset]

- The goal of a reset stylesheet is to reduce browser inconsistencies in things like default line heights, margins and font sizes of headings
- Eric Meyer created a Reset stylesheet for public use, that is in use on millions of websites today

```
html, body, div, span, applet, object, iframe, h1, h2, h3, h4, h5, h6, p, blockquote, pre, a, abbr,
acronym, address, big, cite, code, del, dfn, em, img, ins, kbd, q, s, samp, small, strike, strong,  sub,
sup, tt, var, b, u, i, center, dl, dt, dd, ol, ul, li, fieldset, form, label, legend, table,  caption,
tbody, tfoot, thead, tr, th, td, article, aside, canvas, details, embed, figure, figcaption,  footer,
header, hgroup, menu, nav, output, ruby, section, summary, time, mark, audio, video {
    margin: 0;
    padding: 0;
    border: 0;
    font-size: 100%;
    font: inherit;
    vertical-align: baseline;
}
/* HTML5 display-role reset for older browsers */
article, aside, details, figcaption, figure, footer, header, hgroup, menu, nav, section {
    display: block;
}
```

[DAN.IT EDUCATION]

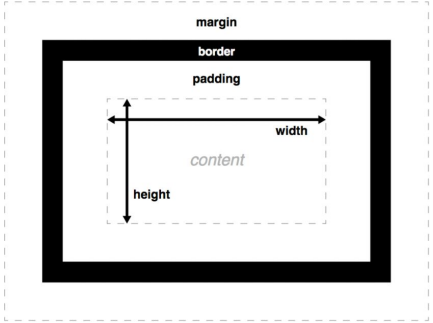
[CSS Reset]

```
body {
    line-height: 1;
}

ol, ul {
    list-style: none;
}

blockquote, q {
    quotes:
    none;
}

    blockquote:before, blockquote:after, q:before, q:after {
        content: '';
        content: none;
    }

table {
    border-collapse: collapse;
    border-spacing: 0;
}
```

[DAN.IT] EDUCATION

CSS Box Model

- All HTML elements can be considered as boxes
- Each box consists of:
  - **Content** - The content of the box, where text and images appear
  - **Padding** - Clears an area around the content. The padding is transparent
  - **Border** - A border that goes around the padding and content
  - **Margin** - Clears an area outside the border. The margin is transparent

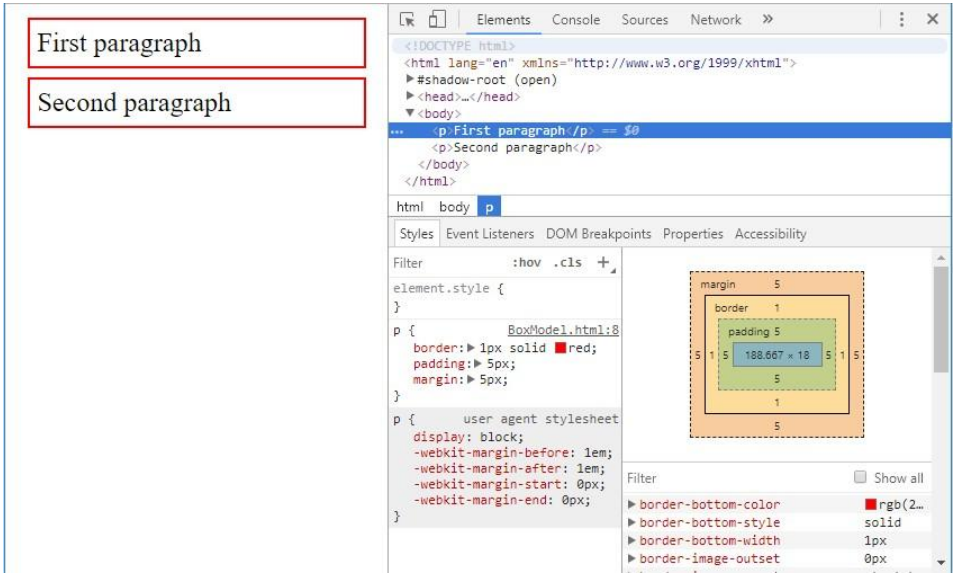



```
width:250px;
padding:10px;
border:5px solid gray;
margin:10px;
```

Let's do the math:
250px (width)
+ 20px (left and right padding)
+ 10px (left and right border)
+ 20px (left and right margin)
= 300px

CSS Box Model In Developer Tools

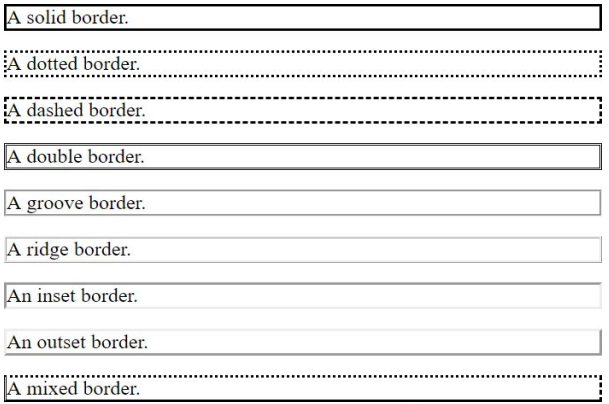
→ You can examine the box model of each individual element on the page by opening up the browser developer tools and clicking on the elements in the DOM inspector



Borders

- The CSS border properties allow you to specify the style, width, and color of an element's border
- The border-style property specifies what kind of border to display
  - This property is required for all the other border properties to take affect
  - It can have from one to four values (for the top/right/bottom/left borders)

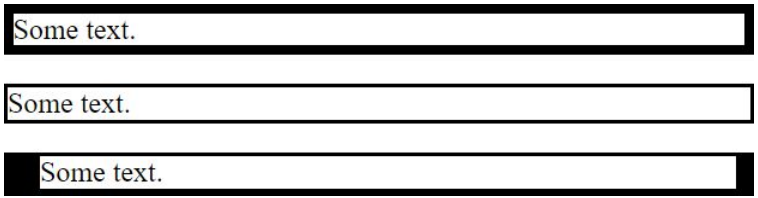
```
p.solid { border-style: solid; }
p.dotted { border-style: dotted; }
p.dashed { border-style: dashed; }
p.double { border-style: double; }
p.groove { border-style: groove; }
p.ridge { border-style: ridge; }
p.inset { border-style: inset; }
p.outset { border-style: outset; }
p.mix { border-style: dotted dashed solid double; }
```



Border Width

- The border-width property specifies the width of the four borders
- The width can be set as a specific size (in px, pt, cm, em, etc.) or by using one of the three pre-defined values: thin, medium, or thick
- The border-width property can have from one to four values (for the top/right/bottom/left borders)

```
p.one {
    border-style: solid;
    border-width: 5px;
}
p.two {
    border-style: solid;
    border-width: medium;
}
p.three {
    border-style: solid;
    border-width: 2px 10px 4px 20px;
}
```



Border Color

- The border-color property is used to set the color of the four borders
- The border-color property can have from one to four values (for the top/right/bottom/left borders)
- If border-color is not set, it inherits the color of the element

```
p.color_one {
    border-style: solid;
    border-color: red;
}

p.color_two {
    border-style: solid;
    border-color: red green blue yellow;
}
```

A solid red border

A solid multicolor border

Border – Individual Sides

- There are also properties for specifying each of the borders (top, right, bottom, and left)
- If the border-style property has four values: **border-style: dotted solid double dashed;**
  - top border is dotted
  - right border is solid
  - bottom border is double
  - left border is dashed
- If the border-style property has two values: **border-style: dotted solid;**
  - top and bottom borders are dotted
  - right and left borders are solid
- If the border-style property has one value: **border-style: dotted;**
  - all four borders are dotted
- The same works with border-width and border-color

[DAN.IT] EDUCATION

Border – Individual Sides

```
p.individual_sides {
    border-top-style: dotted;
    border-right-style: solid;
    border-bottom-style: dotted;
    border-left-style: solid;
}

/* The same as: */
p.individual_sides {
    border-style: dotted solid;
}
```

Border – Shorthand Property

- The border property is a shorthand property for the following individual border properties:
  - border-width
  - border-style (required)
  - border-color

```
p.border {
    border: 5px solid red;
}
```

This property is a shorthand property for border-width, border-style, and border-color.

- You can also specify all the individual border properties for just one side:

```
p.leftBorder {
    border-left: 6px solid red;
    background-color: lightgrey;
}
```

This property is a shorthand property for border-left-width, border-left-style, and border-left-color.

Rounded Borders

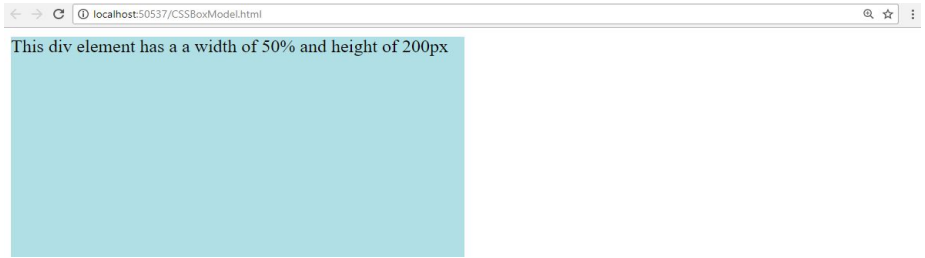
- The border-radius property is used to add rounded borders to an element
  - This property is not supported in IE8 and earlier versions

```
p.normal {
    border: 2px solid red;
}
p.round1 {
    border: 2px solid red;
    border-radius: 5px;
}
p.round2 {
    border: 2px solid red;
    border-radius: 8px;
}
p.round3 {
    border: 2px solid red;
    border-radius: 12px;
}
```

Width and Height

- The width and height properties are used to set the width and height of an element
- They can be specified by:
  - auto (default) - the browser calculates the width and height
  - length values, like px, cm, etc.
  - percent (%) of the containing block
- The width and height properties do not include padding, borders, or margins
  - They set the size of the area inside the element!

```
div {
    width: 50%;
    height: 200px;
    background-color: powderblue;
}
```



Box Sizing

- The box-sizing property (CSS3) defines how the width and height of an element are calculated: should they include padding and borders, or not
- Possible values:
  - content-box (default) - The width and height properties (and min/max properties) includes only the content. Border and padding are not included.
  - border-box - The width and height properties (and min/max properties) includes content, padding and border
  - inherit - Inherits this property from its parent element

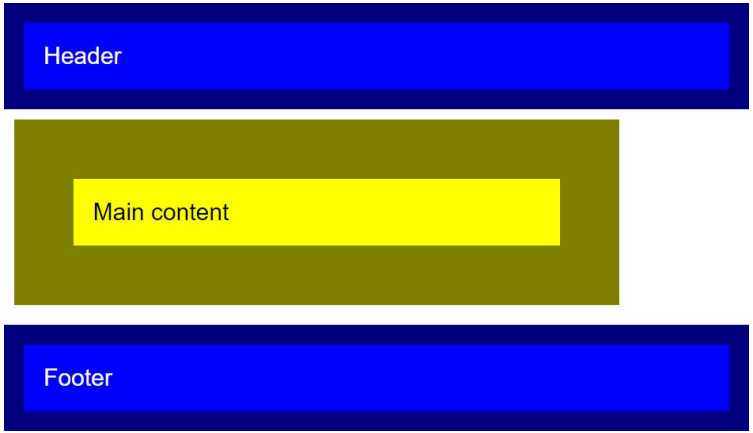
```
div.box {
    box-sizing: border-box;
    width: 50%;
    border: 5px solid blue;
    float: left;
}
```

Exercise (2)

- You are given the following HTML:

```
<div id="wrapper">
    <header>Header</header>
    <main>Main content</main>
    <footer>Footer</footer>
</div>
```

- Use CSS to make the page look like this:



CSS Layout – The display Property

- The display property specifies if/how an element is displayed
- Every HTML element has a default display depending on what type of element it is
  - The default display value for most elements is block or inline
- A block-level element always starts on a new line and takes up the full width available (stretches out to the left and right as far as it can)
  - Examples of block-level elements: , <h1> - <h6>, <p>, <form>, <header>, <footer>, <section>
- An inline element does not start on a new line and only takes up as much width as necessary
  - Examples of inline elements: , <a>, <img>

Override The Default Display Value

- Changing an inline element to a block element, or vice versa, can be useful for making the page look a specific way, and still follow the web standards.
- A common example is making inline <li> elements for horizontal menus:

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

Display a list of links as a horizontal menu:

HTML CSS JavaScript

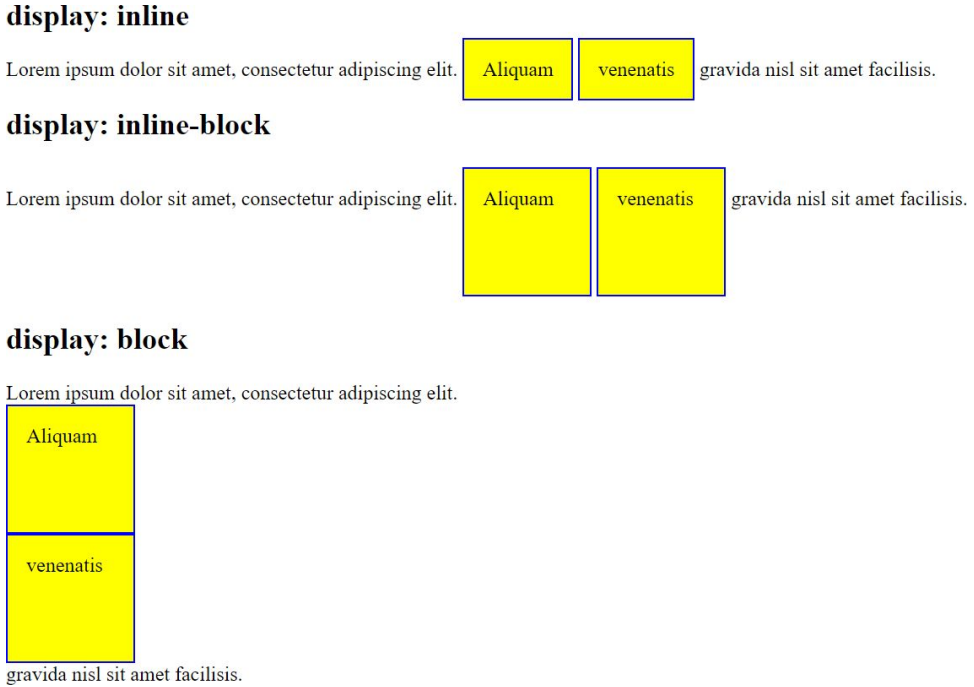
- Setting the display property of an element only changes **how the element is displayed**, NOT what kind of element it is
  - e.g., an inline element with display: block; is not allowed to have other block elements inside it

Inline-Block

- display: inline-block allows to set a width and height on an inline element
- Also, the top and bottom margins/paddings are respected, in contrast to display:inline
- Compared to display: block, the major difference is that display: inline-block does not add a line-break after the element, so the element can sit next to other elements
- The following example shows the different behavior of display: inline, display: inline-block and display: block:

[Inline vs. Inline-Block vs. Block]

```
span.a {
     display: inline;
     width: 70px;
     height: 70px;
     padding: 15px;
     border: 1px solid blue;
     background-color: yellow;
}
span.b {
     display: inline-block;
     width: 70px;
     height: 70px;
     padding: 15px;
     border: 1px solid blue;
     background-color: yellow;
}
span.c {
     display: block;
     width: 70px;
     height: 70px;
     padding: 15px;
     border: 1px solid blue;
     background-color: yellow;
}
```



Hide an Element

- Hiding an element can be done by setting the display property to none
  - The element will be hidden, and the page will be displayed as if the element is not there

```
h1.hidden {
    display: none;
}

<h1>This is a visible heading</h1>
<h1 class="hidden">This is a hidden
heading</h1>
```

- visibility:hidden; also hides an element
  - However, the element will still take up the same space as before

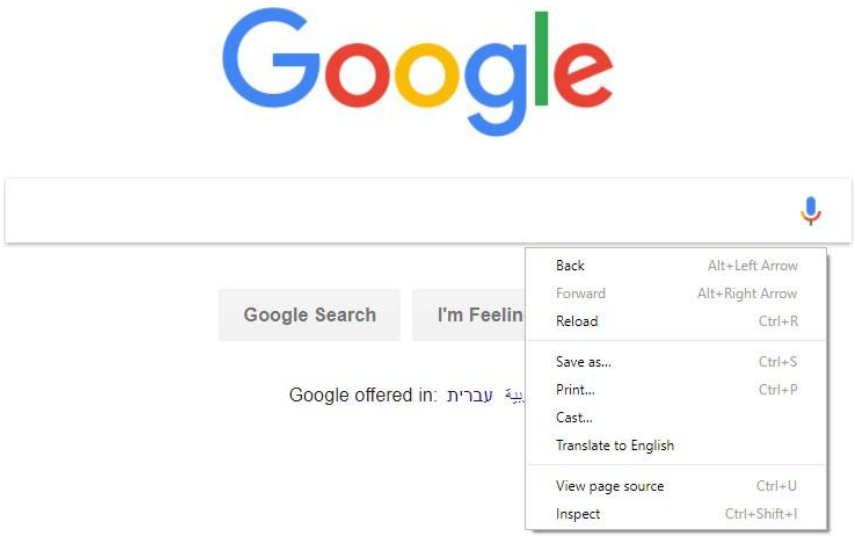
```
h1.hidden2 {
    visibility: hidden;
}
<h1>This is a visible heading</h1>
<h1 class="hidden2">This is a hidden
heading</h1>
```

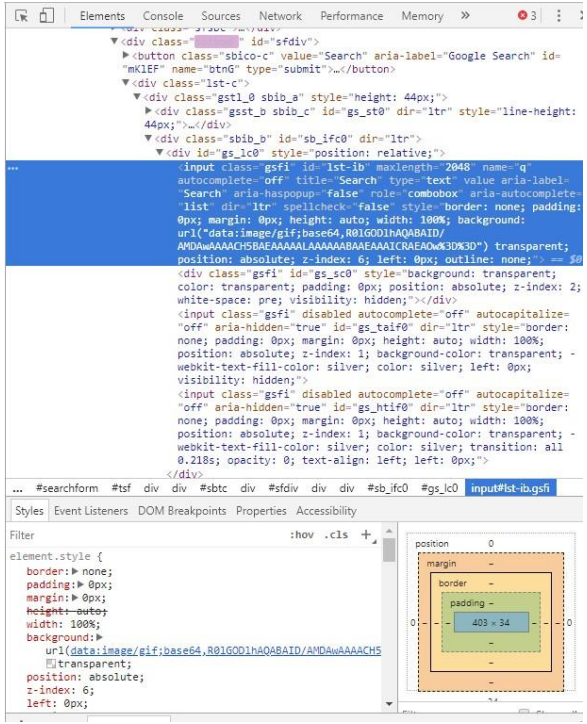
[DAN.IT EDUCATION]

How To View HTML Source?

- Have you ever seen a Web page and wondered "Hey! How did they do that?"
- View HTML Source Code:
  - To find out, right-click in the page and select "View Page Source" (in Chrome) or similar in other browsers. This will open a window containing the HTML source code of the page.
- Inspect an HTML Element:
  - Right-click on an element, and choose "Inspect" or "Inspect Element" to see what elements are made up of (you will see both the HTML and the CSS)
  - You can also edit the HTML or CSS on-the-fly in the Elements or Styles panel that opens

[How To View HTML Source?]





[DAN.IT EDUCATION]

Control questions

1. Why do we need reset.css?
2. What is a block model?
3. How does inline, block and inline-block modes work?
4. What is the purpose of box-sizing property?
5. How can you examine HTML elements in your browser?