Internet Computing

CSS

What is CSS?

- Look at the following for an example of CSS
    - http://example.com/
- Benefits of CSS
    - More readable.
    - It's easier to update and maintain
    - It is possible to completely restyle your website without actually changing the HTML of the page at all
        - https://www.w3schools.com/css/css_intro.asp

Levels of Style Sheets: inline

- Inline
  - stylings are linked to a specific tag.
- Disadvantages:
  - The styling is all mixed up inside our content
  - Makes it not readable
  - difficult to maintain.
- Solution
  - Internal CSS/Document-level CSS

```
General format
        style = "property_1: value_1;
                                property_2:
value_2;
                                 …
                                property_n:
value_n"

Example:
<p style=”color:blue”> This is a text
</p>
```

Levels of Style Sheets: Document-level/internal

- apply to the whole document in which they appear
  - Style sheet appears as a list of rules that are the content of a `<style>` tag
  - Style tag is places in the `<head>` traditionally
  - The <style> tag must include the type attributeset to "text/css"
  - Pairs are separated by semicolons, just as in the value of a `<style>` tag
  - Comments in the rule list:
    - use C comments (/*…*/)

```
General form:
<style type = "text/css">
          rule list
</style>

Form of the rules:
         selector {list of property/values}
Each property/value pair has the form:
          property: value;
```

Levels of Style Sheets: External

- On a separate file, any server on the internet
- Can be applied to any number of documents
- Written as text files with the MIME type `text/css`
- A `<link>` tag is used to specify that the browser is to fetch and use an external style sheet file
- Format of rules is the same as those in the content of a `<style>` tag for document-level style sheets

```
<link rel = "stylesheet"  type = "text/css"
 href = "http://www.wherever.org/termpaper.css">
</link>
```

Precedence of different levels of style sheets

- When more than one style sheet applies to a specific tag in a document, the lowest level style sheet (inline) has precedence
  - In a sense, the browser searches for a style property specification starting with inline, until it finds one (or there isn’t one)

Selector

- Specifies the elements to which the styles apply
- Simple Selectors
  - The selector is a tag name or a list of tag names, separated by commas
- Class Selectors
  - Used to allow different occurrences of the same tag to use different style specifications
  - A style class has a name, which is attached to a tag name
  - The class you want on a particular occurrence of a tag is specified with the class attribute of the tag

```
h1, h3 {
}

p {
}
```

```
p.narrow
{property/value list}
 p.wide {property/value
list}

<p class = "narrow">
    ...
</p>
    ...
<p class = "wide">
    ...
</p>
```

Selector

- Generic Selectors
  - A generic class can be defined if you want a style to apply to more than one kind of tag
  - A generic class must be named, and the name must begin with a period

```
.sale { … }

<h1 class = "sale"> Weekend
Sale </h1>
    ...
<p class = "sale"> … </p>
```

- id selectors
  - An id selector allows the application of a style to one specific element
  - General format
  - `#specific-id {property-value list}`

```
#section14 {
    color:blue
    }

<div>
id=”#section14”>
</div>
```

Classes vs IDs

- An element can have multiple classes associated with it
- An element can have one id
- IDs should be unique
- Classes
  - Type of an element
  - Used to consistently style multiple elements
- IDs
  - Unique name of an element

Contextual Selector

- Select an element by listing one or more of the ancestors of the element
  - Descendant Selector
    - ul ol {property-value list}
      - applies to ol when it is in a ul element
  - Child Selector
    - ul > ol
      - applies to ol when it is a child of a ul element
    - p > h1 > em
      - applies to em when it is the child of an h1 element that is the child of a p element
  - p:first-child, p:last-child, p:only-child for specific children
  - p:empty for no children

Pseudo class

- styles that apply when something happens, rather than because the target element simply exists
- Names begin with colons
  - In contrast to name of style classes and generic classes that starts with a period
- Pseudo classes for styling any element
  - `hover` classes apply when the mouse cursor is over the element
  - `focus` classes apply when an element has focus
- Pseudo class for styling hypertext links
  - `link` classes apply when a link has not been selected
  - `visited` classes apply when a link previously has been selected
- DEMO

Properties in CSS

- There are a large number of properties, arranged in categories:
  - For a full list, check
    - https://developer.mozilla.org/en-US/docs/Web/CSS/Reference
- Most commonly used categories of properties
  - font
  - list
  - Alignment of text
  - margins
  - colors
  - backgrounds
  - borders

Property values format in CSS

- Keywords
  - left, small, …
  - Not case sensitive
- Number Values
  - An integer or decimal number
- Length Values
  - Number values followed immediately by a two-character abbreviation of a unit name
  - Units: px - pixels, in - inches, cm - centimeters, mm - millimeters, pt - points,
  - pc - picas (12 points), em - the value of current font size in pixels, ex - height of the letter ‘x’
  -

Property value format

- Percentage - just a number followed immediately by a percent sign
  - a value relative to the previously used measure for a property
- URL values
  - url(protocol://server/pathname)
  - url(http://www.example.com/bar.gif)
  - url(tetons.jpg)
- Colors
  - Color name
  - rgb(n1, n2, n3)
  - Hex form: #XXXXXX

Property Values

- Many property values are inherited by all descendent elements
  - Inherited
    - Font-size is an example of a property that is inherited
      - By setting the `font-size` in the `<body>` element, the font of the whole document changes
  - NOT inherited
    - `background-color`
    - `margin`
- You can define a property to be inherited or not
  - `div {background-color: inherit;}`

The font properties

- `font-family`: a generic code can be specified as a font family
- `font-size`:
  - small, x-small: Not the same on different browsers
  - 1.2em: Sets to 1.2 times the font size of the parent element
  - Percentage: Adjust the font size relative to the font size of the parent elements
  - `Smaller`: The amount of change is not the same among browsers
  - percentage and em are the best options
- `Font-weight`: degrees of boldness
  - bolder, lighter, bold, normal (could specify as a multiple of 100 (100-900))
- `font` (shorthand)
  - For specifying a list of font properties
    - Order must be: style, weight, size, name(s)

```
font: bolder 14pt Arial Helvetica
```

- `Examples: fonts.html, fonts2.html`

The text properties

- The `text decoration` property
  - `line-through, overline, underline, none`
  - Example: Decoration.html

The text spacing properties

- `letter-spacing` property
  - the amount of space between the letters in words – tracking possible values:
    - normal or any length value
    - Positive length values increase spacing
    - Negative length values decrease spacing
- `word-spacing` property – the amount of space between words
  - Possible values – like those of letter-spacing
- `line-height` property – space between lines – leading
  - Possible values – a number, which is the number of times the font size, or a percentage
- Example: The text_space.html

List Properties

- `List-style-type`
  - Unordered list
    - Bullet can be a disc (default), a square, or a circle
    - Set it on either the <ul> or <li> tag
    - On <ul>, it applies to all items in the list
    - On <li>, list-style-type applies to just that item
    - Could use an image for the bullets in an unordered list
  - Example:
    - Look at the course page for examples

Ordered list

- `list-style-type` can be used to change the sequence values

Property value	Sequence type	First four
decimal	Arabic numerals	1, 2, 3, 4
upper-alpha	Uc letters	A, B, C, D
lower-alpha	Lc letters	A, b, c, d
upper-roman	Uc Roman	I, II, III, IV
lower-roman	Lc Roman	I, ii, ii, iv

- There are several more, including none
- Example:
  - Look at the course page for examples

Alignment of Text

- The `text-indent` property allows indentation
  - Takes either a length or a % value
- The `text-align` property has the possible values
  - left (the default), center, right, or justify
- Sometimes we want text to flow around another element
  - the `float` property
- The `float` property has the possible values, `left, right, and none` (the default)
- If we have an element we want on the right, with text flowing on its left, we use the default `text-align` value (left) for the text and the `right` value for float on the element we want on the right
- Example:
  - Look at the course page for examples

The box model

Border

Margin

padding

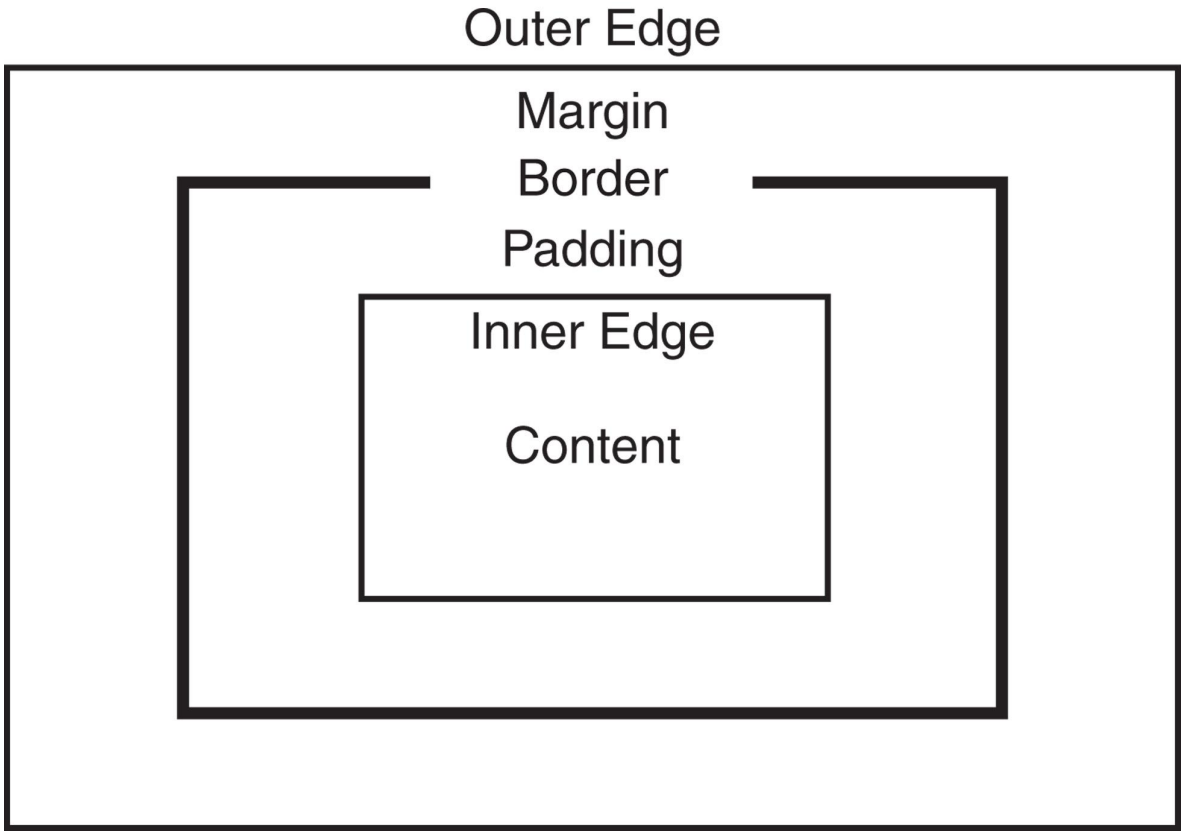


Image taken from [2]

The border

- every element has a `border-style` property
  - Controls whether the element has a border and if so, the style of the border
    - `border-style` values: none, dotted, dashed, and double
    - `border-width` – thin, medium (default), thick, or a length value in pixels
      - border width can be specified for any of the four borders (e.g., `border-top-width`)
    - `border-color` – any color
      - Border color can be specified for any of the four borders (e.g., `border-top-color`)
    - Table borders and table cell borders
      - To get cell borders: `td, th {border: thin solid black}`
      - To get table bordesrs: `table {border: thin solid black}`
  - Example: look at the course page examples

The margin and padding

- `margin`
  - the space between the border of an element and its neighbor element
  - The margins around an element can be set with `margin-left`, etc. -
    - just assign them a length value
- `padding`
  - the distance between the content of an element and its border
  - Controlled by `padding`, `padding-left`, etc.
- Example:
  - marpad.html

Background images

- The `background-image` property
- Repetition can be controlled
  - `background-repeat` property
  - Possible values: `repeat` (default), `no-repeat`, `repeat-x`, or `repeat-y`
  - `background-position` property
    - Possible values: `top`, `center`, `bottom`, `left`, or `right`

colors

- Colors
  - https://htmlcolorcodes.com/
  - https://www.w3schools.com/colors/colors_picker.asp
- Fonts
  - Font-family property
  - Google Fonts
- Example
  - https://www.w3schools.com/code/tryit.asp?filename=FZ9EC1BRJ0FX

The and tags

- : used to style a part of document
  - Used to create document sections (or divisions) for which style can be specified
    - e.g., A section of five paragraphs for which you want some particular style
- : used to style part of a <p> element

```
<style type = "text/css">
     .bigred {font-size: 24pt;
                  font-family: Ariel; color: red}
</style>

<p>
Now is the <span class = "bigred"> best time
</span> ever!
</p>

<p>
     Now is the <span> best time </span> ever!
</p>
```

CSS layout

- How the elements are positioned on the page
- Normal flow:
  - How the block-level elements and inline elements are normally displayed by the browser.
- Block-level elements are each contained on their line
  - Example block-level elements: `<p>, <div>, <h2>, <ul>, and <table>`
- inline elements elements are displayed within lines formed by block-level elements
  - They do not form their own blocks.
  - Example Inline elements: `<text>, <img>`

Block-level elements

Each block exists on its own line and is displayed in normal flow from the browser window's top to its bottom.

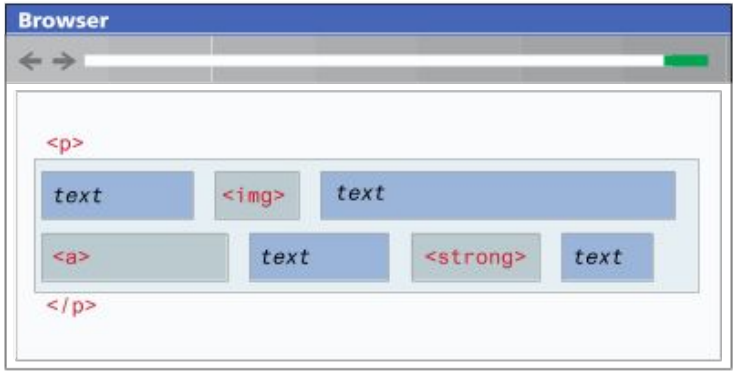
By default each block-level element fills up the entire width of its parent (in this case, it is the <body>, which is equivalent to the width of the browser window).

You can use CSS box model properties to customize, for instance, the width of the box and the margin space between other block-level elements.

Taken from [2]

Inline elements

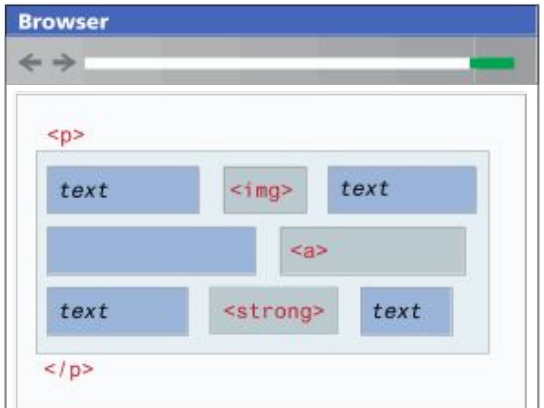
```
<p>
This photo <img src="photo-con.png" alt="..." /> of Conservatory Pond in
<a href="http://www.centralpark.com/">Central Park</a> New York City
was taken on October 22, 2015 with a <strong>Canon EOS 30D</strong>
camera.
</p>
```



Inline content is laid out horizontally left to right within its container.

Once a line is filled with content, the next line will receive the remaining content, and so on.

Here the content of this <p> element is displayed on two lines.

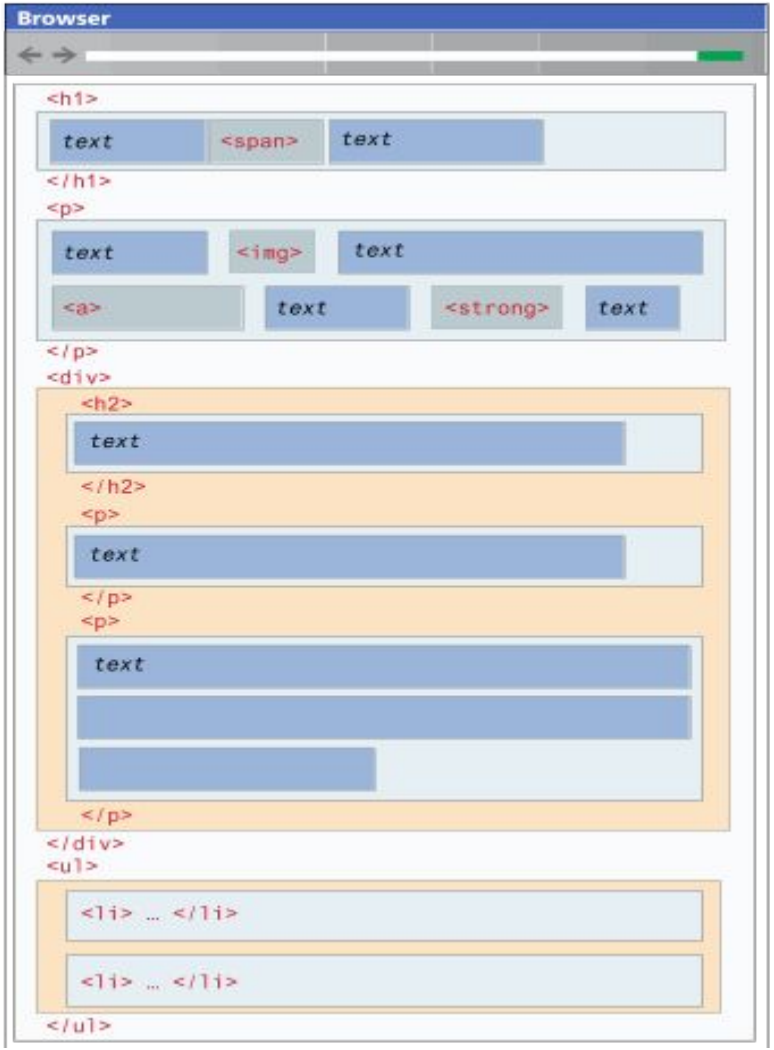


If the browser window resizes, then inline content will be "reflowed" based on the new width.

Here the content of this <p> element is now displayed on three lines.

Taken from [2]

Block-level and inline elements combined



Taken from [2]

Positioning Elements

- `static`
  - The element is positioned according to the normal flow. This is the default.
- `relative`
  - The element is moved relative to where it would be in the normal flow.
- `absolute`
  - Takes the element out of the flow of the page and allows us to move it around relative to the nearest positioned ancestor
    - So basically all other elements position themselves as if the element with absolute position isn't there at all.
  - Example:https://www.w3schools.com/howto/howto_css_image_text.asp

Positioning Elements

- `fixed`:
  - Takes the element out of the flow of the pag (e.g., menu ittem)
  - The element stays in the position
  - a typical example for a fixed positioning might be if you want to have a menu at the top of your page and you want that to appear even as the user scrolls down the page.
- `fixed` **vs** `absolute`
  - `absolute` will move when the page is scrolled, `fixed` will not.

The `z-index` property

- “The `z-index` property specifies the stack order of an element. An element with greater stack order is always in front of an element with a lower stack order.
  Note: z-index only works on positioned elements (position:absolute, position:relative, or position:fixed).”[3]
-
- Note:
  - By adding a `z-index` property to an element, a stacking context is created
    - every child of that element will stack on top of it.
    - To place an element behind its parent, don't create a stacking context on the parent. Use a negative `z-index` though, because the default stack level of the parent is zero

The `float` property

- Displaces an element out of its position in the normal flow
  - An element can be floated to the left or floated to the right .
  - it is moved all the way to the far left or far right of its containing block and the rest of the content is “reflowed” around the floated element
- Floated elements come one after another, but everything that is floated left will be aligned to the left of the screen

The `clear` property

- If you combine floated and non-floated element, then
- Everything that is not floated is going to put itself in the flow of the page.
- To avoid that we can use `clear` property
- If we use `clear` property on an element, it is positioned so that it will not not be adjacent to floated element

The `clear` property values

- `left`
  - The left-hand edge of the element cannot be adjacent to another element.
- `right`
  - The right-hand edge of the element cannot be adjacent to another element.
- `both`
  - Both the left-hand and right-hand edges of the element cannot be adjacent to another element.
- `none` The element can be adjacent to other elements.

References

1. Programming the World Wide Web, 8th edition
2. Fundamentals of Web Development, 2nd edition
3. https://www.w3schools.com/