Lecture

HTML: Style Sheets

What are We Talking About Today?

- **Style Sheets & CSS**
  - Separate content and markup from design
  - Inline, internal, and external CSS
- **CSS Syntax**
  - Selector {property: value;}
  - Classes, IDs, tag selectors
- **Advanced Features**
  - Specificity, cascading rules
  - CSS3: transitions, animations, media queries, filters, variables, color science, calculations, layers, nesting, advanced selectors, accessibility features

Style Sheets

- Starting with HTML version 4.x there was an effort to **separate** the specification of **style** (presentation) from the specification of **content and markup**
- Style sheets are the mechanism HTML assumes will be used to specify
  - the amount of white space between text or between lines,
  - the amount lines are indented,
  - the colors used for the text and the backgrounds
  - the font size and style of text
  - the precise position of text and graphics
  - How front matter (preface, figure list, title page, and so forth) should look
  - How all or individual sections should be laid out in terms of space (for example, two newspaper columns, one column with headings having hanging heads, and so forth)

CSS

- CSS stands for Cascading Style Sheets
  - It is not a markup language. It is a list of Rules
- CSS is like JSON Objects, with keyword: value pairs
  - CSS keywords are already defined though

Version	Year
CSS 1	1996
CSS 2	1998
CSS 2.1	2011
CSS 3 modules	2012-2022
CSS3	ongoing

Style Sheet Languages

- These notes use the style language "Cascading Style Sheets" ([CSS1] and [CSS2]), abbreviated CSS, and described in:

  http://www.w3.org/TR/REC-CSS1 (level 1)

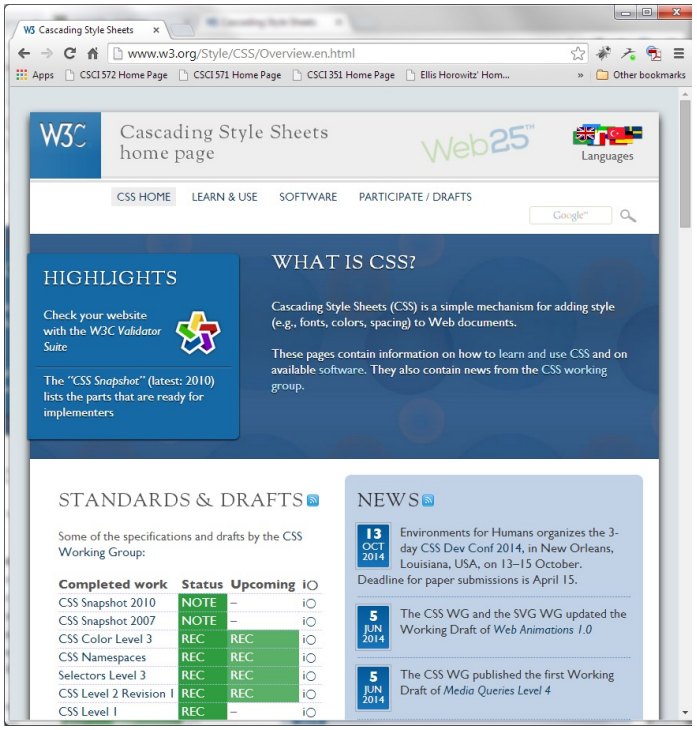
  http://www.w3.org/TR/REC-CSS2 (level 2)

  http://www.w3.org/TR/CSS21/ (level 2 Revision 1)

  http://www.w3.org/Style/CSS/current-work (Level 3, in process)

  http://www.w3.org/Style/CSS/Overview.en.html (see Status)

- There are other style sheet languages, e.g., XSL defined at http://www.w3.org/Style/XSL

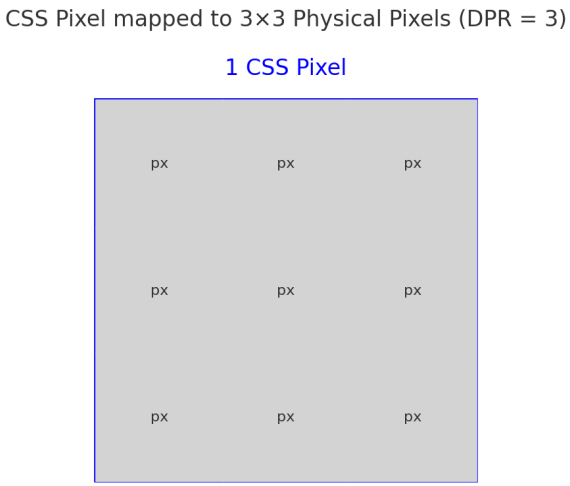


CSS Pixels vs Physical Pixels

- **CSS Pixels**
  - Abstract units used in CSS/HTML for layout and styling
  - Stay consistent across devices regardless of screen density
  - Mapped by browser to physical pixels via device pixel ratio (DPR)
- **Physical Pixels**
  - Actual hardware dots of light on the display
  - Fixed by the device resolution (e.g.,
- **Device Pixel Ratio (DPR)**
  - Defines how many physical pixels represent 1 CSS pixel
  - Example: iPhone 15 Pro → DPR ≈ 3 (1 CSS pixel = 3×3 device pixels)
  - Ensures consistent sizing across devices while improving sharpness
- **CSS viewport**
  - Browser-defined rectangular area, measure in CSS pixels, to render the page
  - iPhone 15 Pro (1179 × 2556), CSS viewport: 430 × 932

CSS Pixels vs Physical Pixels (cont'd)

Concept	CSS Pixel	Physical Pixel
Definition	Abstract unit in web layout	Actual hardware pixel on screen
Fixed or Scales?	Scales depending on DPR	Fixed by hardware
Example (iPhone 15 Pro Max)	430 × 932	1179 × 2556
Relation	1 CSS px = DPR × DPR physical px	Smallest dot on the screen



Expressing Style Within HTML

- CSS can be included in three ways
  1. **Inline** in an HTML element through the **style attribute**
  2. In the **<style> element**, contained in the <head> of an HTML document
  3. In an external file that is included in an HTML document in the **<link> tag**
- Combining style information from multiple sources, called "cascading"
- There is a defined order of precedence where the definitions of a style element conflict
- These HTML extensions permit
  - flexible placement of style information
  - independence from any particular style sheet language

Simple Example

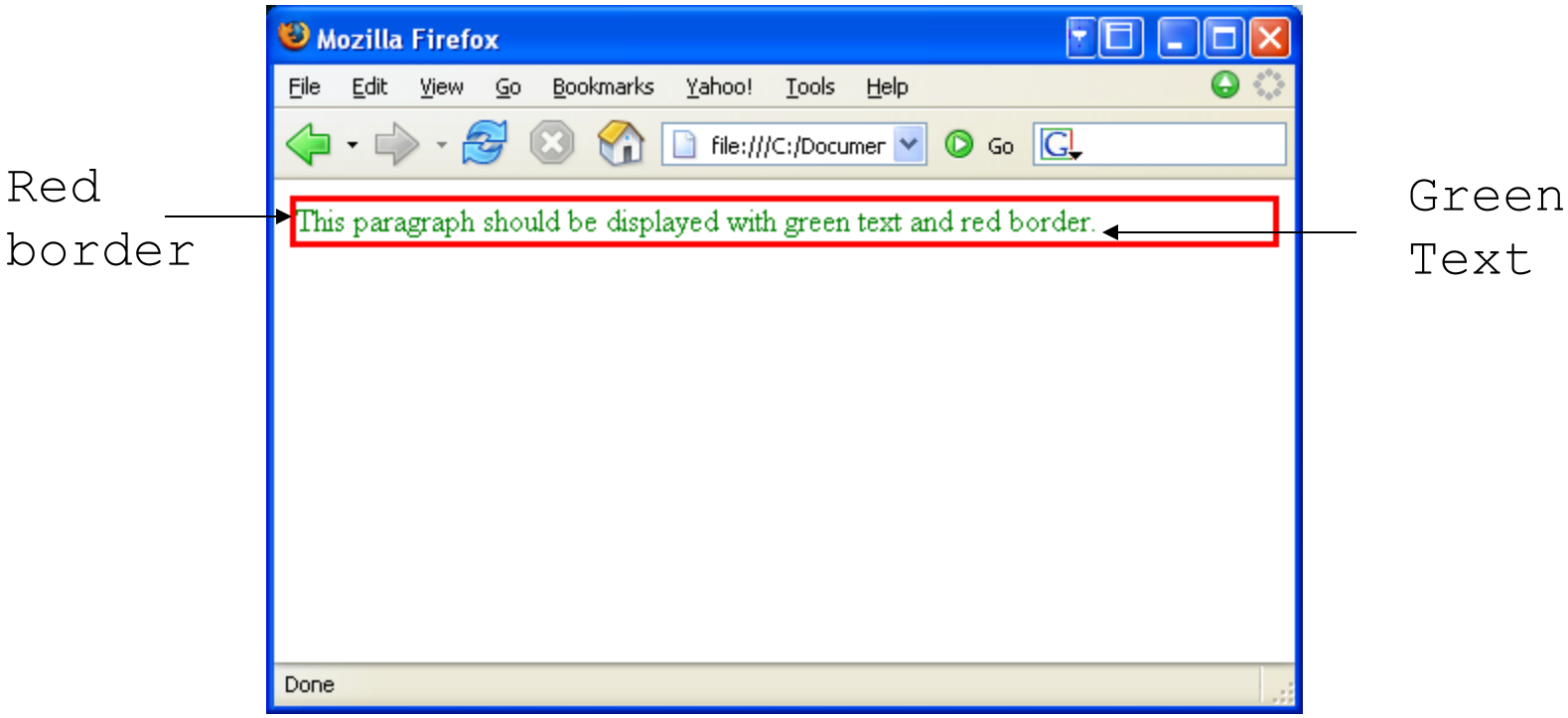
- Suppose a file, special.css contains 3 lines that sets the text color of a paragraph to green and surrounds it with a solid red border:

```
P.special {
  color: green;
  border: solid red;
}
```

- Authors may link this style sheet to their source HTML document with the LINK element:

```
<HTML>
  <HEAD>
    <LINK href="special.css" rel="stylesheet" type="text/css">
  </HEAD>
  <BODY>
    <P class="special">This paragraph should be displayed with
green text and red border.
  </BODY>
</HTML>
```

Browser Output



Different ways to specify a color

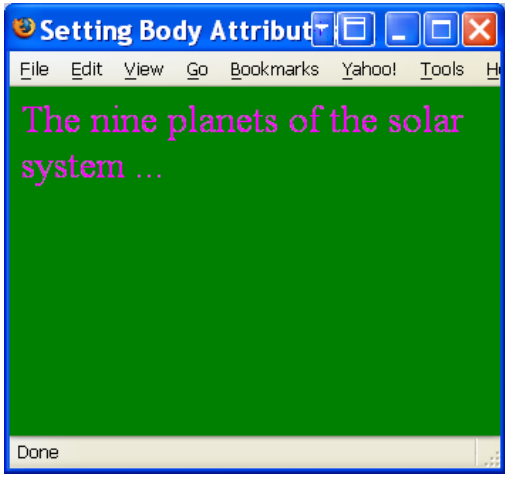
1. RGB HEX (eg. *#f00 - red*, *#ff0000* - red)
2. RGB function
   `rgb(255, 0, 0)` or `rgb(255 0 0)` or `rgb(100% 0% 0%)` - red
   `rgb(255 0 0 / 0.4)` or `rgb(255 0 0 / 40%)` - with 60% opacity
   ! Color functions *rgba()* and *hsla()* are considered legacy
3. RGB-like functions:
   rgb(), hsl(), hwb(), lab(), lch() - pretty old and not accurate
   oklch(), **ok**lab() - modern and self-reflective functions (fixed lch/lab)
   https://evilmartians.com/chronicles/oklch-in-css-why-quit-rgb-hsl
4. Color function (color-space based)
   `color(sRGB 1 0 0)` - full red in sRGB
   `color(display-p3 1 0 0)` - full red in Display-P3
   All supported color spaces: sRGB, Linear-light sRGB, Display P3, A98 RGB, ProPhoto RGB, ITU-R BT.2020-2, and CIE XYZ.
5. Relative colors
   (usually used with css vars to calculate design tokens, like hover/active colors, etc)
   `color: oklch(from var(--base-color) calc(l - 20) c calc(h + 10))`
   take some color, make 20% darker ^ ^ rotate hue by 10deg
6. Color aliases (safe colors, legacy)
7. Over 140 color names standardized by W3C

Option 1: Inline style Attribute

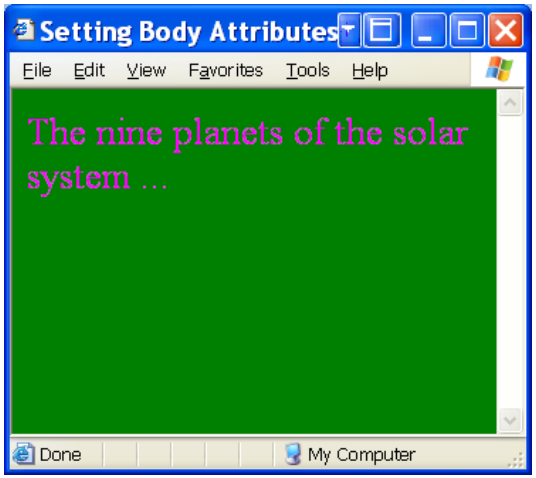
- Using inline style settings to set the font size, background and color of text

```
<HTML>

<HEAD>
  <TITLE>Setting Body Attributes</TITLE>
</HEAD>

<BODY style="font-size: 20pt; background: green; color: fuchsia">
  The nine planets of the solar system ...
</BODY>

</HTML>
```

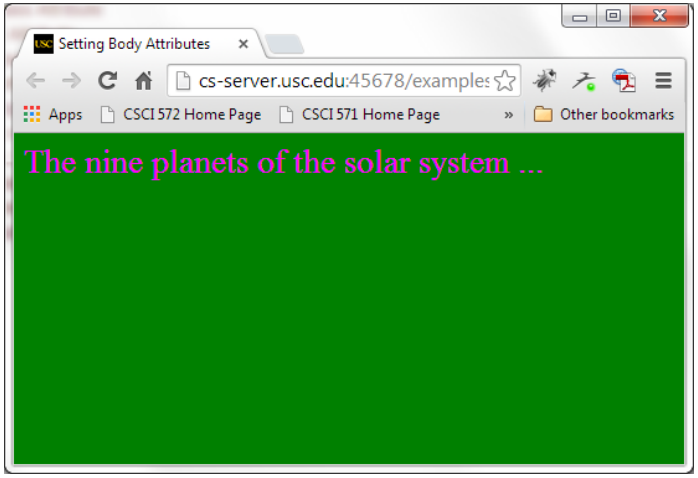
Browser Output



Firefox



Internet Explorer



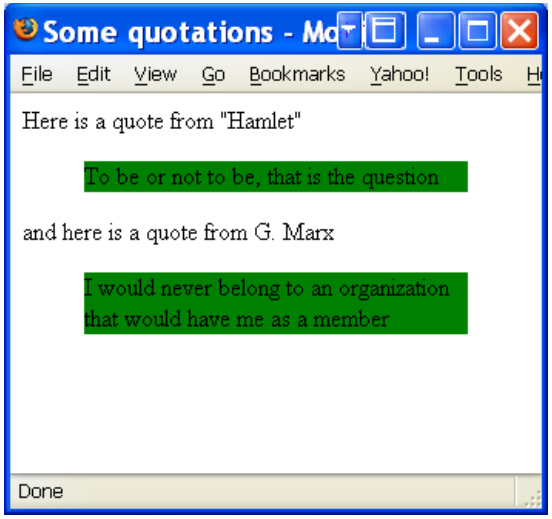
Chrome

Option 2: Using the <STYLE> Element

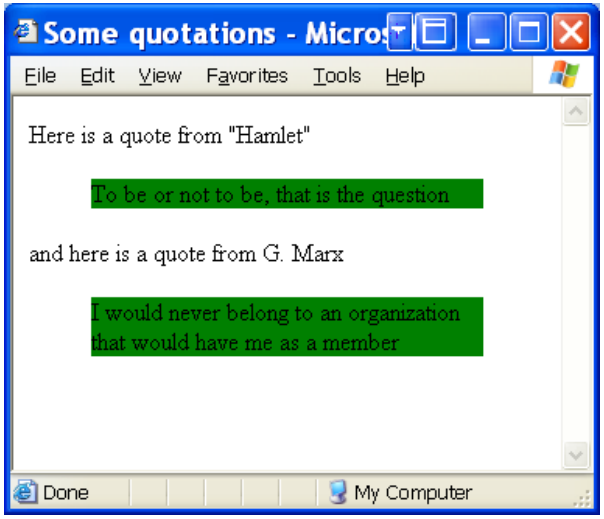
- **<STYLE>** element is placed in the <HEAD>
- Example using <STYLE> element to assign a green background to all QUOTATIONS with class named example

```
<HTML>
<HEAD>
  <TITLE>Some quotations</TITLE>
  <STYLE type="text/css">
    BLOCKQUOTE.example { background: green }
  </STYLE>
</HEAD>
<BODY>
  Here is a quote from "Hamlet"
  <BLOCKQUOTE class="example" id="example-1">
    To be or not to be, that is the question
  </BLOCKQUOTE>
  and here is a quote from G. Marx
  <BLOCKQUOTE class="example" id="example-2">
    I would never belong to an organization that would have me as a member
  </BLOCKQUOTE>
</BODY>
</HTML>
```

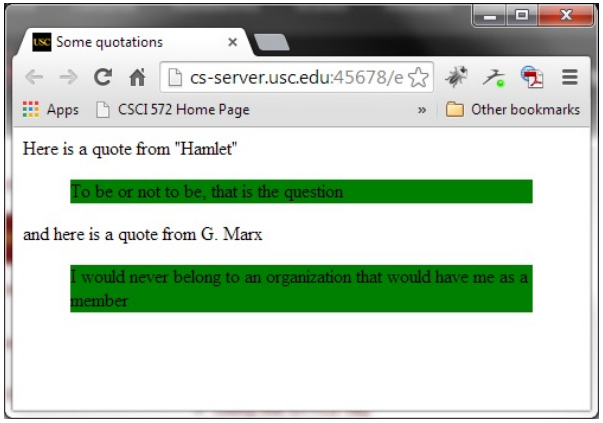
Browser Output



Firefox



Internet Explorer



Chrome

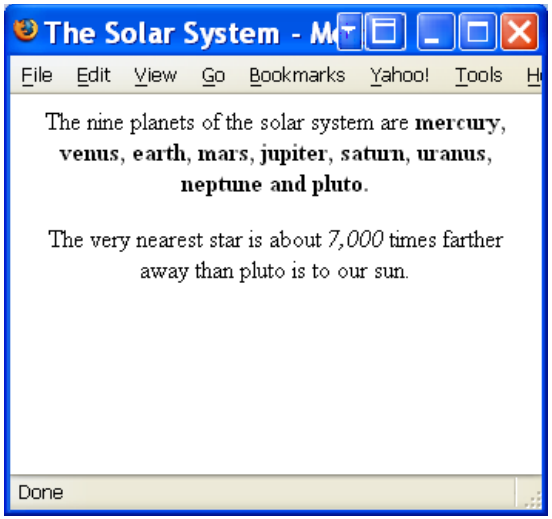
Example - Center Entire Document

```
<HTML>
<HEAD>
  <TITLE>The Solar System</TITLE>
  <STYLE type="text/css">
    BODY { text-align: center }
  </STYLE>
</HEAD>

<BODY>
  <P>The nine planets of the solar system are <B>mercury, venus, earth,
     mars, jupiter, saturn, uranus, neptune and pluto.</B></P>
  <P>The very nearest star is about <I>7,000</I> times farther away than
     pluto is to our sun.</P>
</BODY>
</HTML>
```

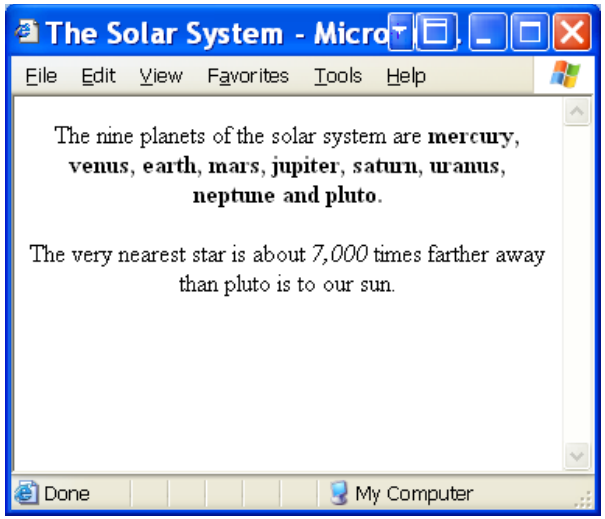
Note <CENTER> element is deprecated, removed in HTML5

```
<style>p {text-align:center;}</style>
```

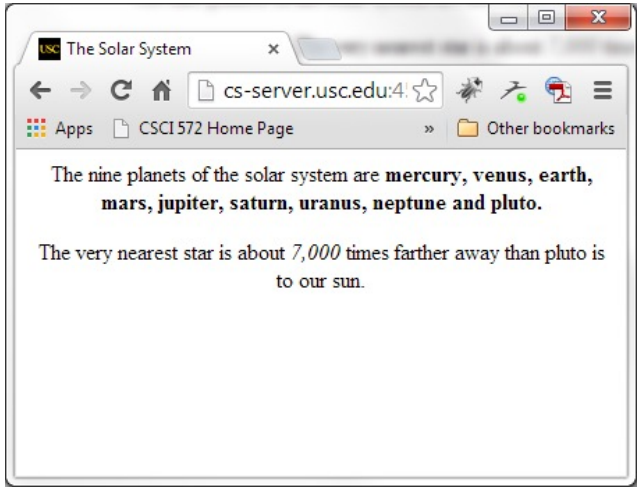
Browser Output



Firefox



Internet Explorer



Chrome

Option 3: Selectors (. #)

- The **CLASS attribute** assigns a name to one or more elements
- The **ID attribute** also can be used in a similar way.
- CLASS and ID can be used as a selectors of style properties

- Ex1: `P.redtext {color: #FF0000}`

would be applied when <P> contains the CLASS name

```
<P CLASS="redtext">some text</P>
```

- Ex2: `.redtext{color: #FF0000}`

applies to all elements that use CLASS=redtext

- Context sensitive selectors

```
STRONG { color: #00FF00}
```

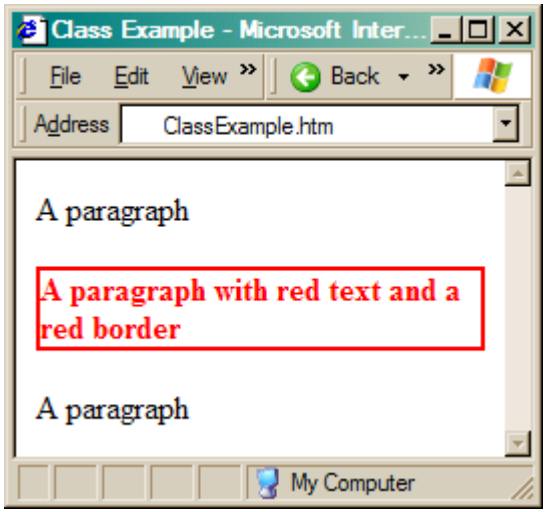
defines all occurrences of <STRONG> as green, but

Class Attribute

- The **class attribute** is used for optional styles

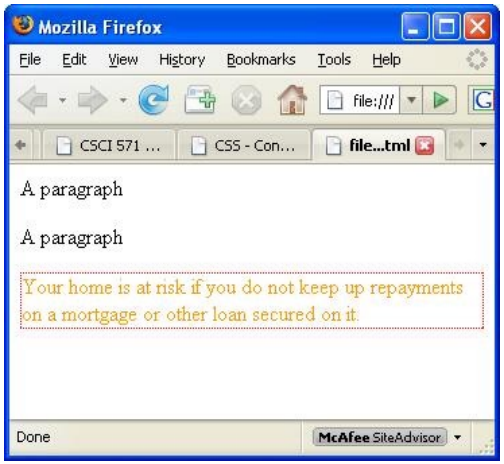
```
<head>
     <style>
          .Warning { font-weight: bold; color: red;
                                  border:2px solid red;}
     </style>
</head>
<body>
     <p>A paragraph</p>
     <p class="Warning">A paragraph with
          red text and a red border</p>
     <p>A paragraph</p>
</body>
```

dot character



Id Attribute

- The **id attribute** is also used for optional styles
  - but it can only be used ***ONCE*** in the entire document
- Normally used for major document sections
  - Header, Footer, TopNav, LeftNav,
  - Content, etc.



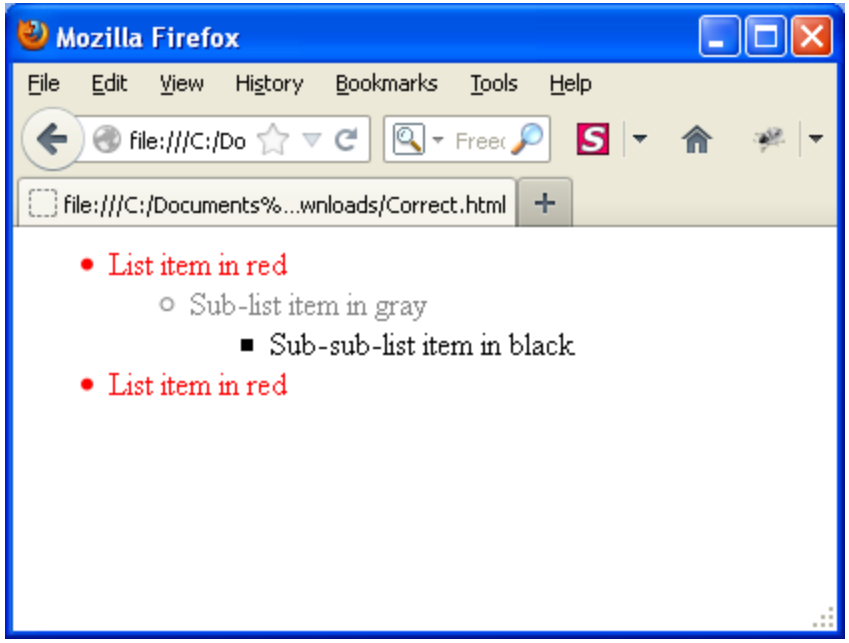

```
<head>
     <style>
          #Disclaimer {  font-size:medium;
                         color: #ff9900;
                               border:1px dotted red;}
     </style>
</head>
<body>
     <p>A paragraph</p>
     <p>A paragraph</p>
     <p id="Disclaimer">Your home is at risk if you
          do not keep up repayments on a mortgage
          or other loan secured on it.</p>
</body>
```

 # character

Specific Styles

- Styles aimed at specific tags

```
<head>
  <style>
    ul { color: red; }
    ul ul { color: gray; }
    ul ul ul { color: black }
  </style></head>
<body>
  <ul>
    <li>List item in red</li>
    <ul>
      <li>Sub-list item in gray</li>
      <ul>
        <li>Sub-sub-list item in black</li>
      </ul>
    </ul>
    <li>List item in red</li>
  </ul>
</body>
```

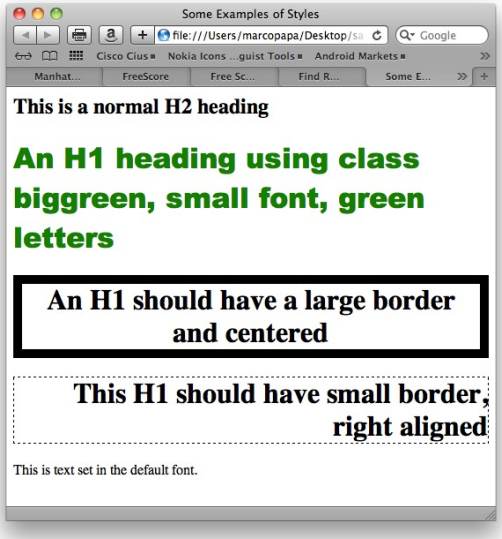


More <STYLE> Element Examples

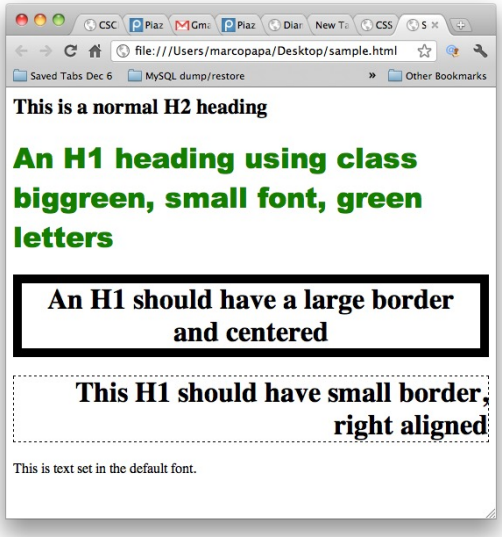
- Review
  - **CLASS rule preceded by "." and applied to multiple elements**
  - **ID rule preceded by "#" and applied to single elements**
  - **Values assigned to ID and CLASS are case sensitive**

```
<HTML><HEAD><TITLE>Some Examples of Styles</TITLE>
<STYLE type="text/css">
  .biggreen {font-family: "Arial Black"; font-size: 30; color: green;}
  H1.myh1 {border-width: 10; border-style: solid; text-align: center}
  #myid {border-width: 1; border-style: dashed; text-align: right}
</STYLE></HEAD>
<BODY>
  <H2>This is a normal H2 heading</H2>
  <H1 CLASS="biggreen">An H1 heading using class biggreen, small font,
     green letters</H1>
  <H1 class="myh1">An H1 should have a large border and centered</H1>
  <H1 id="myid">This H1 should have small border, right aligned  </H1>
  This is text set in the default font.
</BODY>
</HTML>
```

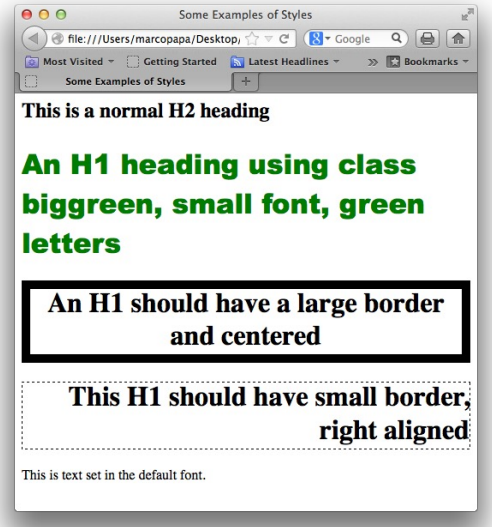
Browser Output



Safari



Chrome



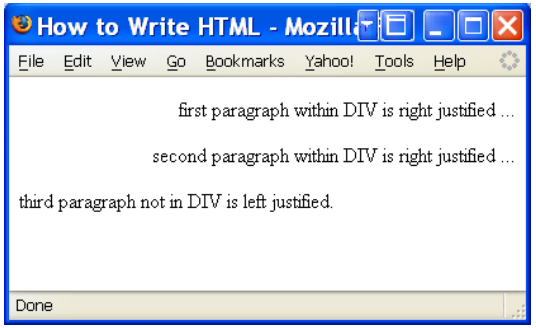
Firefox

HTML Block Tags

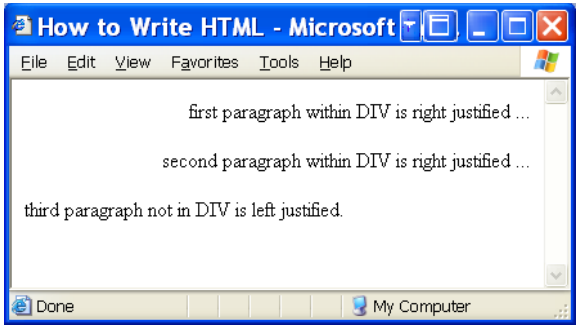
- <DIV> and <SPAN> tags have no initial presentation properties
  - **exception**, line break before and after a <DIV> tag
  - **<SPAN>** applies to **inline** elements (example: <b>)
  - **<DIV>** applies to **block** elements (example: <p>)
- With CSS, properties such as text-align are "inherited" from the parent element

```
<HTML><HEAD><TITLE>How to Write HTML</TITLE>
<STYLE type="text/css">
  DIV.mypars {text-align: right}
</STYLE>
<BODY>
  <DIV class="mypars">
    <P>first paragraph within DIV is right justified ...
    <P>second paragraph within DIV is right justified ...
  </DIV>
  <P>third paragraph not in DIV is left justified.
</BODY></HTML>
```

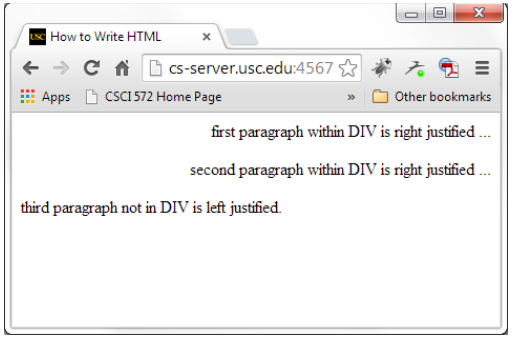
Browser Output



Firefox



Internet Explorer



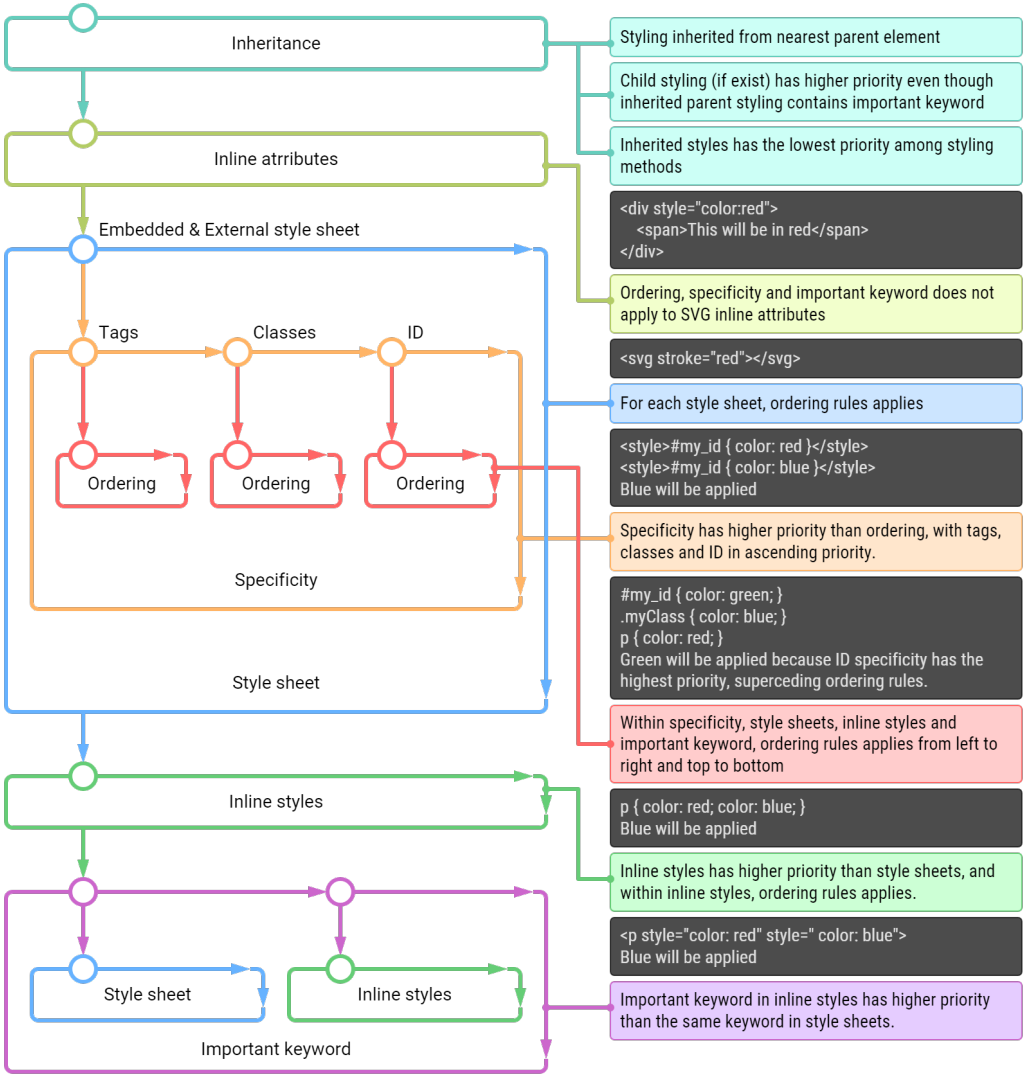
Chrome

Precedence (specificity) of Style Settings

- The more precise a specification is, the higher the precedence, i.e., the more likely it is used
  - H1.mypars {text-align: center} [highest]
  - .biggreen {text-align: center} [next]
  - H1 { text-align: center} [lowest]
- a style for tag.class has higher precedence than one for .class, which has higher precedence than a style for the tag itself
- styles defined using a "style" attribute (inline) have highest precedence
- styles defined using <STYLE> element have next highest precedence
- styles defined in a separate file, e.g., special.css, have lowest precedence
- **Precedence** is guided by the **CSS cascading order**:

https://www.w3.org/TR/CSS22/cascade.html#cascading-order

Precedence guide



https://vecta.io/blog/definitive-guide-to-css-styling-order

Summary

- **Cascade**
  - At a very simple level this means that the **order of CSS rules matter;** when two rules apply that have equal specificity the one that comes **last** in the CSS is the one that will be used.

- **Specificity**
  - Specificity is how the browser decides which rule applies **if multiple rules have different selectors** but could still apply to the same element.

- **Inheritance**
  - Some CSS property values set on parent elements are **inherited by their child elements,** and some aren't.

- See Tutorial at:

https://developer.mozilla.org/en-US/docs/Learn/CSS/Building_blocks/Cascade_and_inheritance

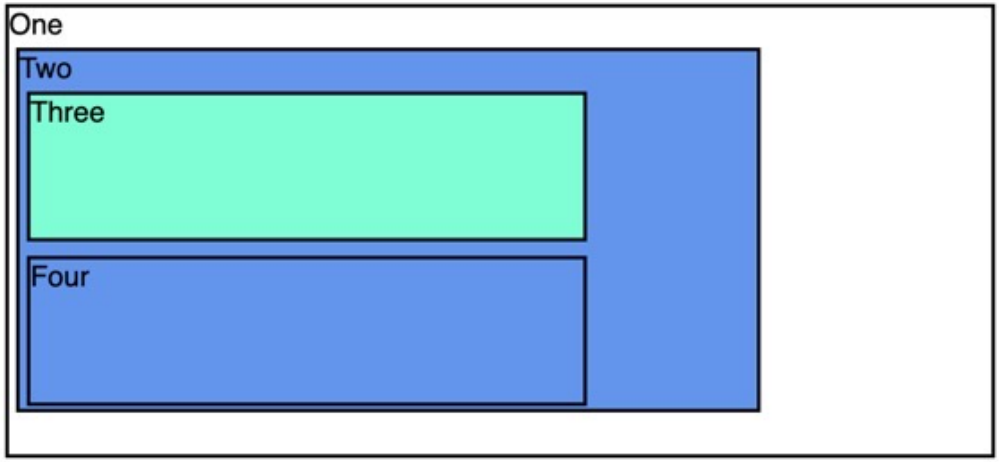
CSS custom properties (variables)

An approach to create dynamic maintainable style systems – light/dark themes, coordinating different sources, getting rid of "magic" values.

- We can define a **variable** and **reference** it in css values
- https://developer.mozilla.org/en-US/docs/Web/CSS/Using_CSS_custom_properties
- These variables use the same nesting inheritance principles
- Value changes are immediately reflected in all referencing styles
- Can hold a value of any css type (colors, sizes, font-families, …)
- Define variable globally:
  ```
  :root { --accent-color: darkblue; --gap: 1rem; }
  ```
- We can use "get with fallback" function:
  ```
  body { padding: var(--may-exist, 2rem); }
  ```
- We can update value (takes effect in element subtree):
  ```
  div.secondary {--accent-color: var(--secondary-color, crimson) }
  ```
- We can declare variables:
  ```
  @property --logo-color {
      syntax: "<color>";
      inherits: false;
      initial-value: #c0ffee;
    }
  ```
- We can get/set/update variables from JS:
  ```
  element.style.setProperty("--my-var", jsVar + 4);
  ```
- Differ from css preprocessor variables (SASS, LESS, etc...) – they are not "baked" into code – they exist and can change in the runtime.

CSS Custom properties example

```
<style>
  div    { background-color: var(--box-color, white) }
  .two   { --box-color: cornflowerblue}
  .three { --box-color: aquamarine }
</style>

<div class="one">
  <p>One</p>
  <div class="two">
    <p>Two</p>
    <div class="three">
      <p>Three</p>
    </div>
    <div class="four">
      <p>Four</p>
    </div>
  </div>
</div>
```



CSS Custom properties example

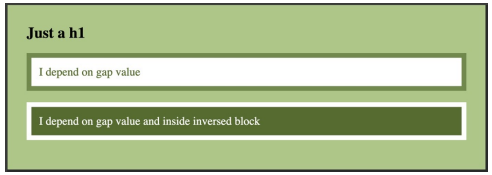
```
<style>
:root { //html
  --accent-color: darkolivegreen;
  --light-color: white;
  --gap: 1rem;
}
body {
  --text-color: var(--accent-color);
  --bg-color: var(--light-color);
  padding: var(--non-existent-variable, 2rem);
  background-color: oklch(from var(--text-color) calc(l + 0.3) c h);;
}

div.inverse {
  --text-color: var(--light-color);
  --bg-color: var(--accent-color);
}

p {
  color: var(--text-color);
  background-color: var(--bg-color);
  border: 10px solid oklch(from var(--text-color) calc(l + 0.1) c h);
  font-size: calc(var(--gap) * 1.5);
  padding: var(--gap);
}
</style>

<body>
<div>
  <h1>Just a h1</h1>
  <p>I depend on gap value</p>
  <div class="inverse">
   <p>I depend on gap value and inside inversed block</p>
  </div>
</div>
</body>
```

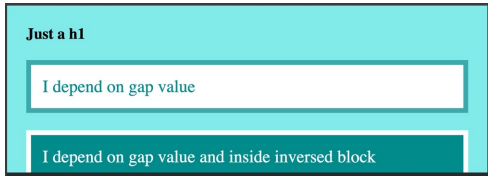
Make 30% lighter

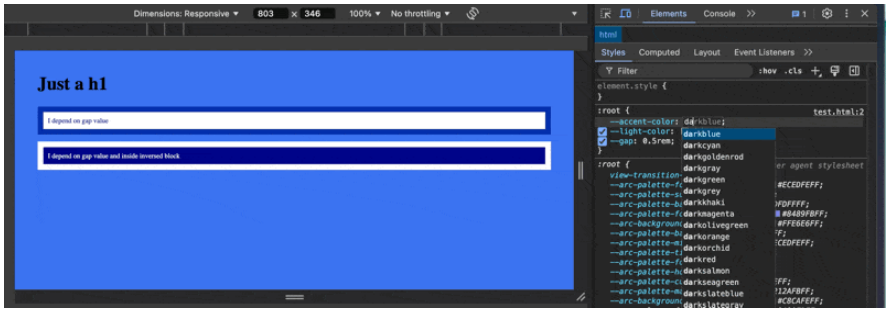
```
--accent-color: darkcyan
--gap:          1.5rem
```

Make 10% lighter

```
--accent-color: crimson
--gap:          0.5rem
```







Composite Styles

- Many styles can be combined in one selector:

```
font: normal small-caps bold small/2em Verdana,
Arial, Helvetica, sans-serif;
```

- Or you could specify:

```
font-family: Verdana, Arial, Helvetica, sans-serif;
font-size:small;
font-style:normal;
font-variant:small-caps;
font-weight:bold;
line-height:2em;
```

- Common uses include:
  - Font
  - Background
  - Margin & Padding

The DOCTYPE Directive

- One of the most important tags on your page
- Instructs modern browsers to work in 'standards compliant mode'
  - Your web page will look the same in ***all*** browsers
  - Browsers turn off their proprietary extensions
  - Fonts are rendered in the same way
    - For example, **font-size: small**, is rendered the same size on all browsers
- **For HTML 5:**

```
<!DOCTYPE html>
```

- **For HTML 4.01**

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
```

Style Sheet Media Types

- Enable authors to create documents for different media types
- **Example: all H1 displayed on a projector are in blue; when printed they are centered**

```
<HEAD>
<STYLE type=text/css media=projection>
 H1 {color:blue}
</STYLE>
<STYLE type=text/css media=print>
 H1{text-align:center}
</STYLE>
```

- **Example: add sound effects to anchors for speech output**

```
<STYLE type=text/css media=aural>
 A {cue-before:url(bell.aiff); cue-after:url
(dong.wav)}
</STYLE></HEAD>
```

Style Sheet Media Types

- Used in CSS3 for **media queries**
- See Responsive Design later in the course
- **Example: Target specific physical characteristics of device.**

```
<link rel="stylesheet" type="text/css" media="screen
and (max-device-width: 480px)" href="min.css" />
```

- **Example: two equivalent pair of media queries**

```
<style>
...
@media all and (min-width:500px) { ... }
@media (min-width:500px) { ... }
</style>
```

Pseudo Elements and Pseudo Classes

- **pseudo elements** and **pseudo classes** are ways of assigning style properties independent of the document tree
- pseudo-classes
  - **:link** – a normal, un-visited link
  - **:visited** – a link the user has visited
  - **:hover** - a link when the user mouses over it
  - **:active** - a link the moment it is clicked
- Example, given the style definition

```
<body><html><head>
<style>
a:link {color:#FF0000;} /* unvisited link (red)*/
a:visited {color:#00FF00;} /* visited link (green)*/
a:hover {color:#FF00FF;} /* mouse over link (pink)*/
a:active {color:#0000FF;} /* selected link (blue)*/
</style></head><body>
```

- See

  https://www.w3schools.com/css/css_pseudo_classes.asp

  https://www.w3schools.com/css/css_pseudo_elements.asp

Pseudo Elements and Pseudo Classes (cont'd)

- Look up the meaning of these other pseudo elements and pseudo classes

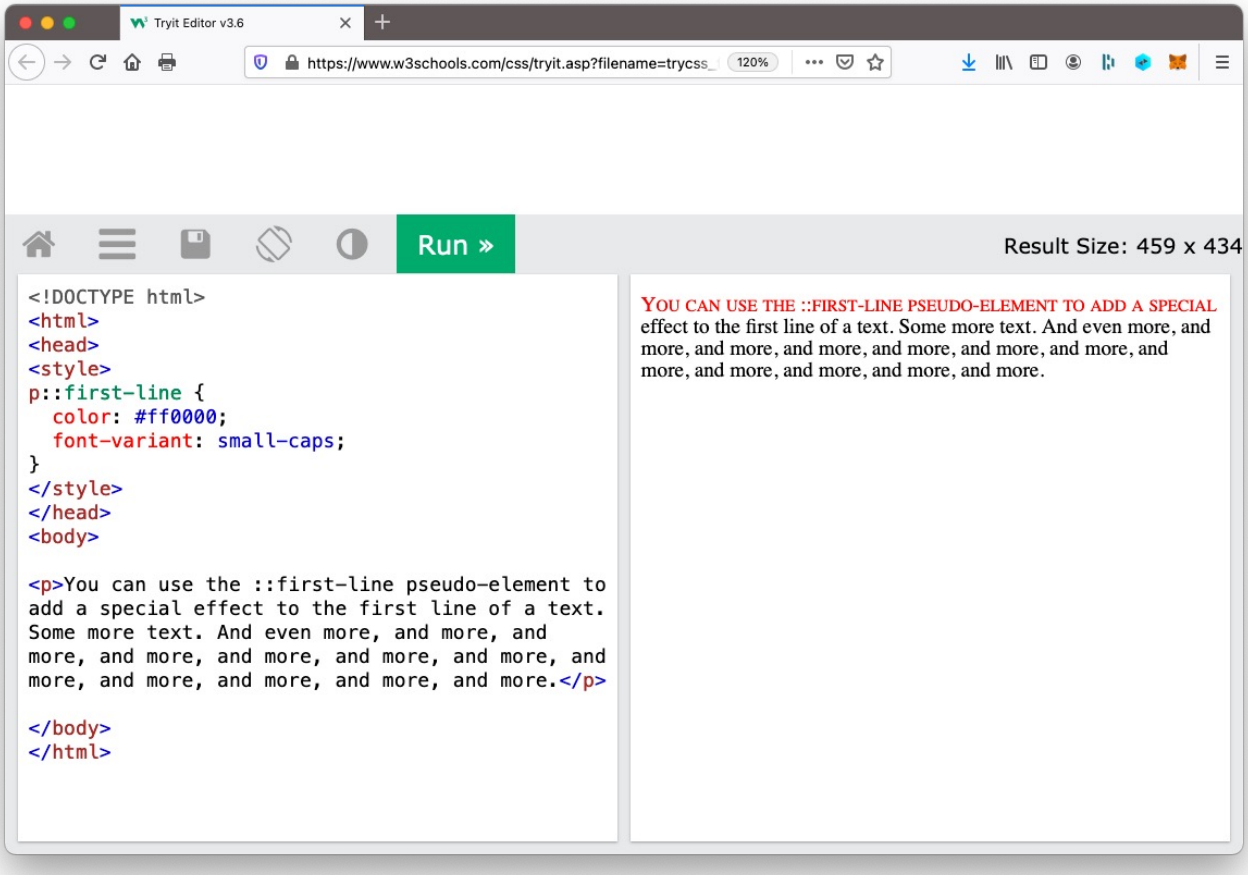
http://www.w3.org/TR/CSS2/selector.html#pseudo-elements

- pseudo **classes**
  - **:first-child**, Selects every <p> elements that is the first child of its parent
  - **:hover**, Selects links on mouse over
  - **:active**, selects the active link
  - **:focus**, selects the input element which has the focus
  - **:lang**, selects every <p> element with a lang attribute
- pseudo **elements**
  - **:first-line**, add a special style to the first line of a text
  - **:first-letter**, add a special style to the first letter of a text
  - **:before**, to insert some content before the content of an element
  - **:after**, to insert some content after the content of an element

Pseudo Elements Example

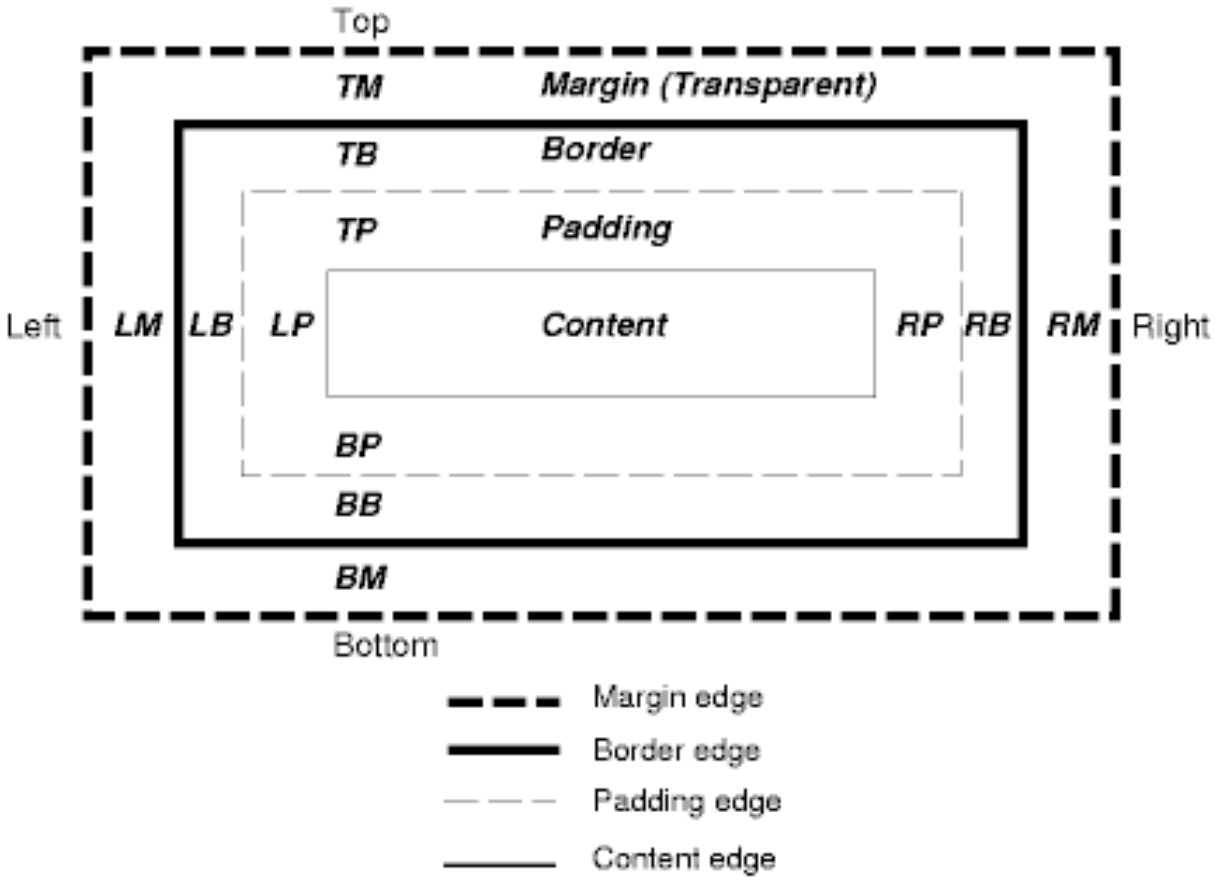
- From W3Schools

  https://www.w3schools.com/css/tryit.asp?filename=trycss_firstline



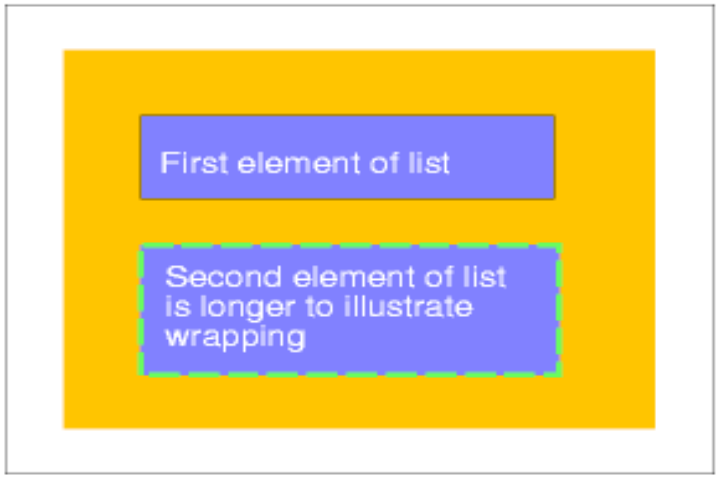
Box Model

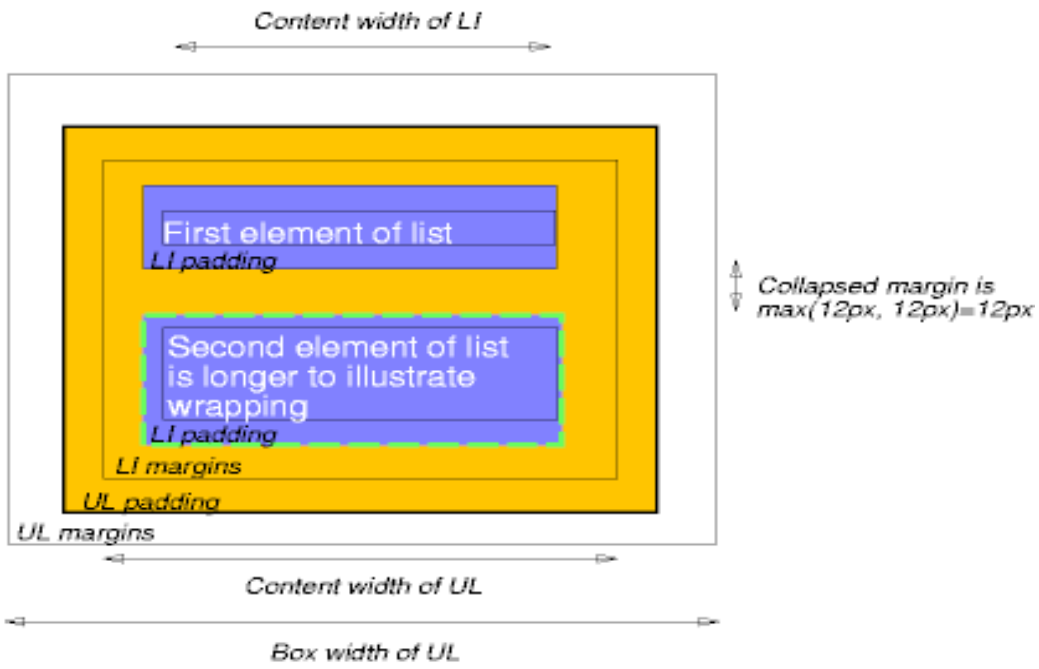
Each box has a ***content*** *area* (e.g., text, an image, etc.) and optional surrounding ***padding***, ***border***, and ***margin*** areas; the size of each area is specified by properties defined below. The following diagram shows how these areas relate, and the terminology used to refer to pieces of margin, border, and padding



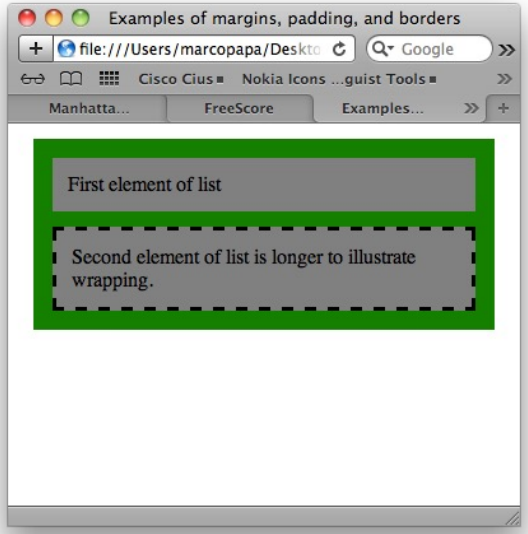
Example of Margins, Padding, Borders

```
<HTML><TITLE>Examples of margins, padding, and borders</TITLE>
<STYLE type="text/css">
UL { background: green;
     margin: 12px 12px 12px 12px;
     padding: 3px 3px 3px 3px; /* No borders set */ }
LI { color: black; /* text color is black */
     background: gray; /* Content, padding will be gray */
     margin: 12px 12px 12px 12px;
     padding: 12px 0px 12px 12px; /* Note 0px padding right */
     list-style: none /* no glyphs before a list item */
     /* No borders set */ }
LI.withborder { border-style: dashed;
     border-width: medium; /* sets border width on all sides */
     border-color: black; } </STYLE> </HEAD>
<BODY>
  <UL>
    <LI>First element of list
    <LI class="withborder">Second element of list is longer to illustrate
    wrapping.
  </UL>
</BODY>
</HTML>
```

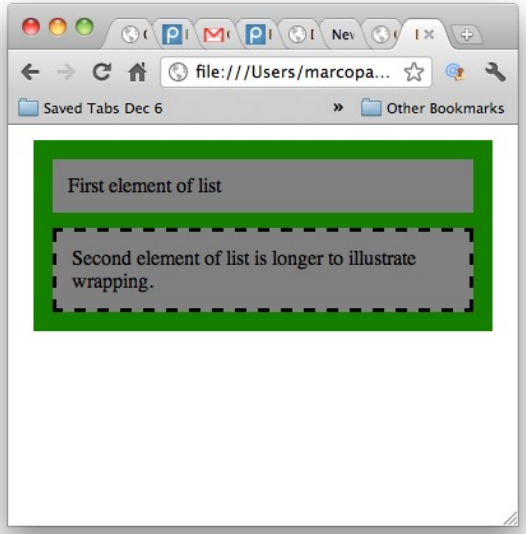




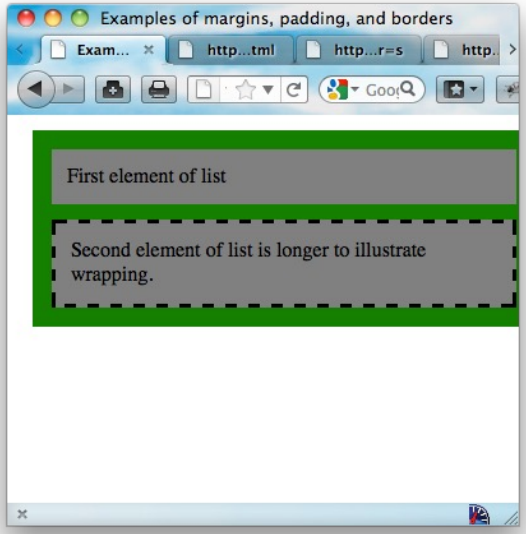
Browser Output



Safari



Chrome



Firefox

https://csci571.com/examples/css/margins.html

CSS Vendor Prefixes

- CSS vendor prefixes or CSS browser prefixes are a way for browser makers to add support for new CSS features during a testing and experimentation period.
- Browser prefixes are used to add new features that may not be part of a formal specification and to implement features in a specification that hasn't been finalized
- The CSS browser prefixes are:
  - Android: **-webkit-**
  - Chrome: -webkit-
  - Firefox: **-moz-**
  - Internet Explorer: **-ms-**
  - iOS: -webkit-
  - Opera: **-o-**
  - Safari: -webkit-
- E.g., before HTML 5, to set a rounded corner on a box one would have to write

```
-moz-border-radius: 10px 5px
-webkit-border-top-left-radius: 10px;
-webkit-border-top-right-radius: 5px;
-webkit-border-bottom-right-radius: 10px;
-webkit-border-bottom-left-radius: 5px;
border-radius: 10px 5px;
```

Style Sheets Are Pervasive

- espn.com
- cbsnews.com
- **microsoft.com**
- Style sheets are often used for "branding" & for changing the look-and-feel
- **Use Firefox Web Developer Inspector, Chrome Developer Tools or Safari Web Inspector to view CSS**

Reset CSS

- A **CSS Reset** is a short, often compressed (minified) set of CSS rules that *resets* the styling of all HTML elements to a consistent baseline.
- Every browser has its own default 'user agent' stylesheet, that it uses to make unstyled websites appear more legible.
  - For example, most browsers by default make links blue and visited links purple, give tables a certain amount of border and padding, apply variable font-sizes to H1, H2, H3 etc. and a certain amount of padding to almost everything.
- The goal of a reset stylesheet is to reduce browser inconsistencies in things like default line heights, margins and font sizes of headings, and so on.
- Reset styles quite often appear in CSS frameworks
- See http://www.cssreset.com/ for several actual code examples

One Sample of css reset

```
/* http://meyerweb.com/eric/tools/css/reset/
   v2.0 | 20110126  License: none (public domain)*/
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;}
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;}
```

Style Sheet Examples

Style Sheet File

- Assume the style rules are stored in the file **mystyle.css**
- These rules re-define the <H1-H5> tags, set background to gray, alter the <P>, <UL>, <LI> tags, and name some color, font and text styles

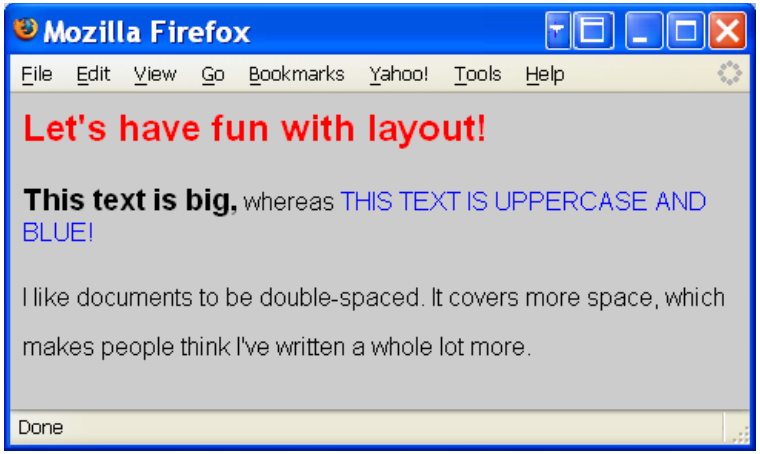
```
h1, h2, h3, h4, h5 { color: red }
body { background-color: #cccccc; font-family: Arial,
Helvetica, sans-serif }
p { line-height: 200% }
ul li { font-size: 70% }
.red { color: red }
.green { color: green }
.blue { color: blue }
#big { font-size: 120% }
#upper { text-transform: uppercase }
```

Example - Using the Style Sheet

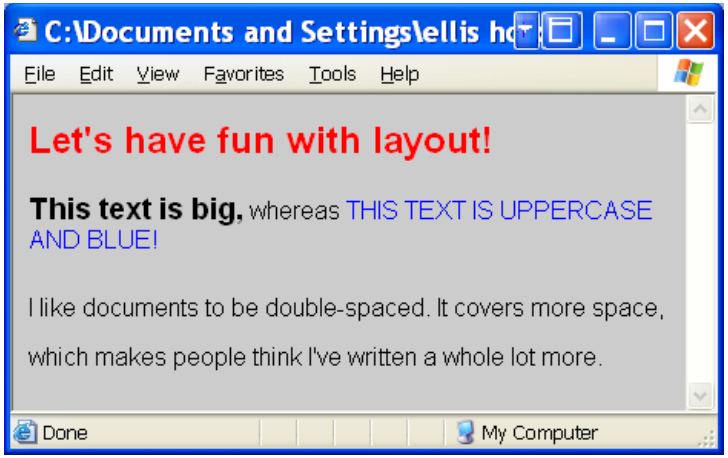
- This example links to an external style sheet and refers to named selectors to produce different colored text and line spacing

```
<HTML>
<HEAD>
  <LINK rel="stylesheet" href="mystyle.css" type="text/css">
</HEAD>
<BODY>
  <H2>Let's have fun with layout!</H2>
  <STRONG id="big">This text is big,</STRONG> whereas
  <SPAN id="upper" class="blue">this text is uppercase and
blue!</SPAN>
  <P>
    I like documents to be double-spaced. It covers more space,
    which makes people think I've written a whole lot more.
  </P>
</BODY>
</HTML>
```

Browser Output



Firefox



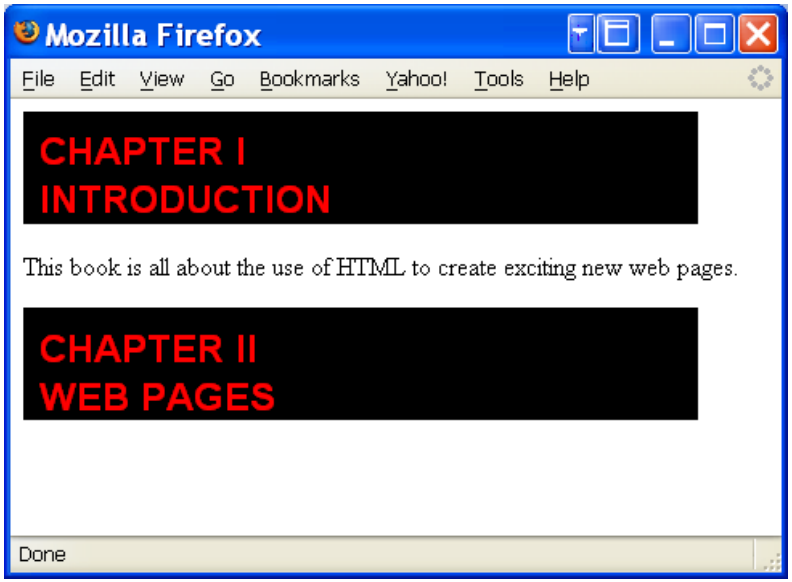
Internet Explorer

Example - Headlines with Graphic Backgrounds

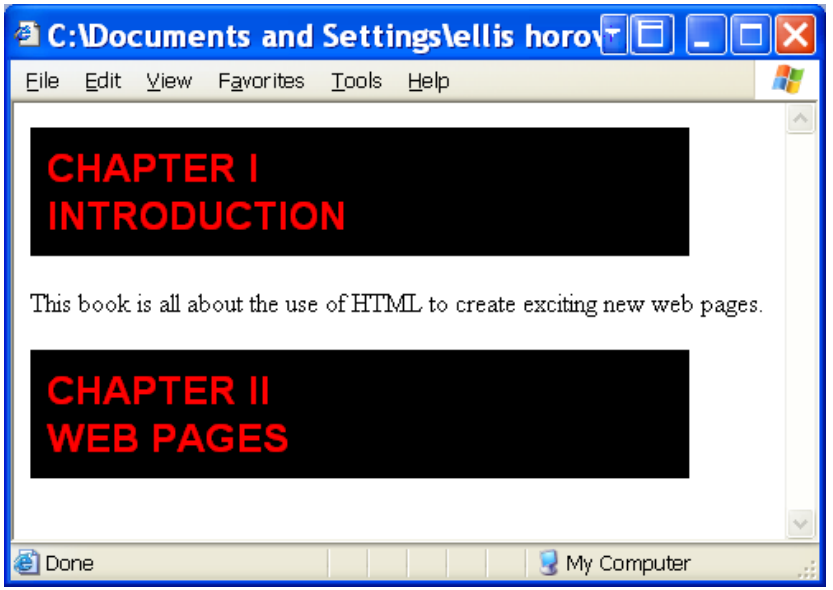
- Using CSS background property, one can create different titles using the same background

```
<HTML>
<HEAD><STYLE TYPE=text/css>
H1 {background:black url(black5.gif) no-repeat; font:bold 18pt
helvetica, sans-serif; color:white; height:50px; width:400px;
padding:10px;}
</STYLE></HEAD>
<BODY>
  <H1>
    <Font color=#FF0000>CHAPTER I<BR>INTRODUCTION</FONT>
  </H1>
  <P>This book is all about the use of HTML to create
     exciting new web pages.
  <H1>
    <Font color=#FF0000>CHAPTER II<BR>WEB PAGES</FONT>
  </H1>
</BODY>
</HTML>
```

Browser Output



Firefox



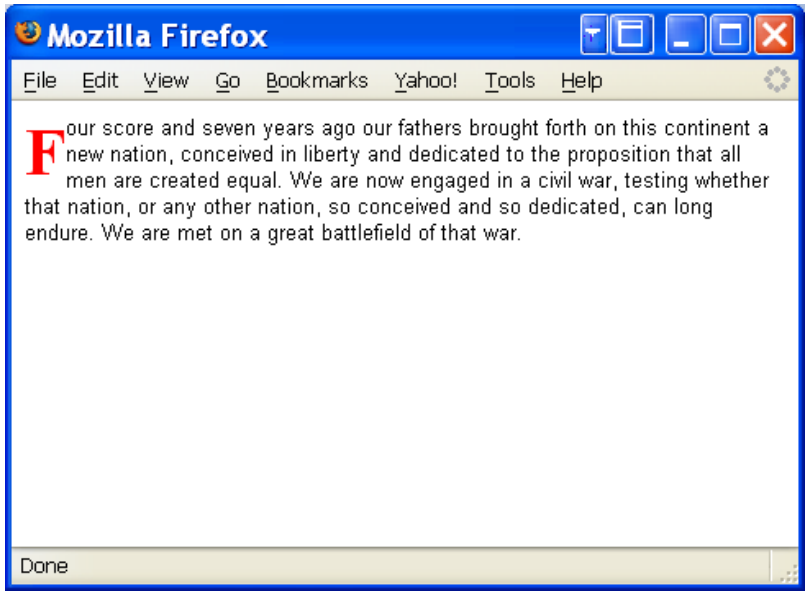
Internet Explorer

Example – Creating Drop Caps

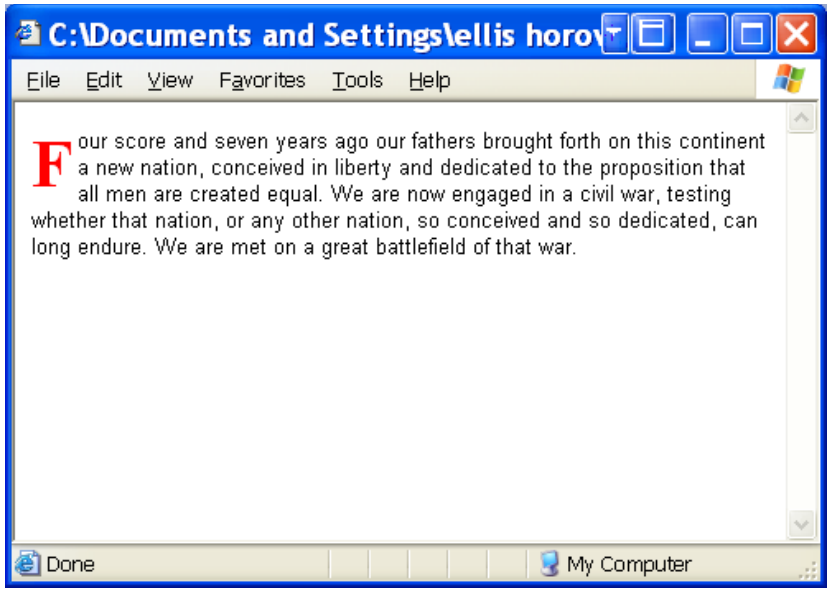
- A traditional form of book style

```
<HTML><HEAD>
<STYLE TYPE=text/css>
P {font: normal 10pt helvetica, arial, sans-serif;}
.dropcap {font:bold 300% times, serif; color:red;
float:left;}
</STYLE></HEAD>
<BODY>
<P><SPAN CLASS="dropcap">F</SPAN> our score and seven years
ago our fathers brought forth on this continent a new
nation, conceived in liberty and dedicated to the
proposition that all men are created equal. We are now
engaged in a civil war, testing whether that nation, or any
other nation, so conceived and so dedicated, can long
endure. We are met on a great battlefield of that war.
</BODY></HTML>
```

Browser Output



Firefox

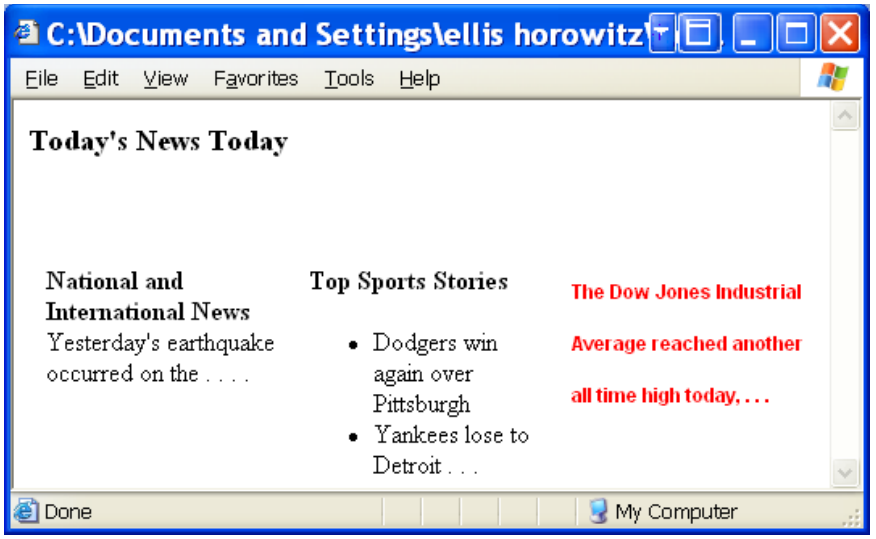


Internet Explorer

Creating Multiple Columns

```
<HTML><HEAD><STYLE TYPE="text/css">
#column1 {position : absolute; top:.5in; left: .1in;
width:1.5in; font:12pt/14pt time,serif;}
#column2 {position : absolute; top:.5in; left: 1.8in;
width:1.5in; font:12pt/14pt time,serif;}
#column3 {position : absolute; top:.5in; left: 3.5in;
width:1.5in; font: bold 9pt/24pt helvetica, sans-serif; color:red;}
.relElement {position: relative; margin: 10px;}
</STYLE></HEAD>
<BODY>
  <H3>Today's News Today</H3>
  <DIV CLASS="relElement">
    <SPAN ID="column1"><b>National and International News</B>
      Yesterday's earthquake occurred on the . . . . </SPAN>
    <SPAN ID="column2"><B>Top Sports Stories</B><BR>
      <UL><LI>Dodgers win again over Pittsburgh
          <LI>Yankees lose to Detroit . . . </UL></SPAN>
    <SPAN ID="column3">The Dow Jones Industrial Average reached another
      all-time high today, . . . <SPAN>
  </DIV>
</BODY></HTML>
```

Browser Output



Firefox

Internet Explorer

Note: Obsolete in HTML 5. Replaced by semantic elements

CSS3 Additional Features

- The W3C has announced the creation of four new modules for CSS Level 3.
- The modules add entirely new functionality and do not extend any previous CSS Level 1 or Level 2 functionality.
- They are based on proposals from Apple's WebKit team, and the current Working Drafts and Recommendations are available at the following URLs:

http://www.w3.org/TR/css3-transitions/ (WD 2023)
http://www.w3.org/TR/css3-transforms-1/ (2D/3D CR 2023)
http://www.w3.org/TR/css3-mediaqueries/ (**REC**)
http://www.w3.org/TR/css3-namespace/ (**REC**)
http://www.w3.org/TR/css3-selectors/ (**REC**)
http://www.w3.org/TR/css3-color/ (**REC**)

CSS 3 New Features

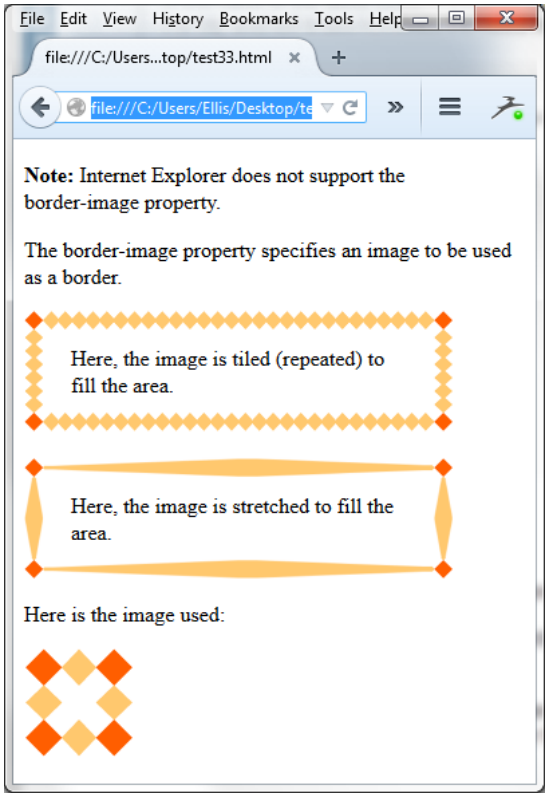
- The new CSS3 features can be organized according to the following categories
  - Borders
  - Backgrounds
  - Text Effects
  - Fonts
  - 2D Transforms
  - 3D Transforms
  - Transitions
  - Animations
  - Multiple Columns
  - User Interface

CSS3 Features supported by All modern Browsers

- Define "modern browsers": **Edge, Opera 10+, Firefox 3.5+, Chrome, Safari 3+**
- box-sizing
- border-radius
- box-shadow
- RGBA Colors
- HSLA Colors
- Multiple Backgrounds
- background-clip
- background-origin
- background-size
- Transforms
- Media Queries

Border Images

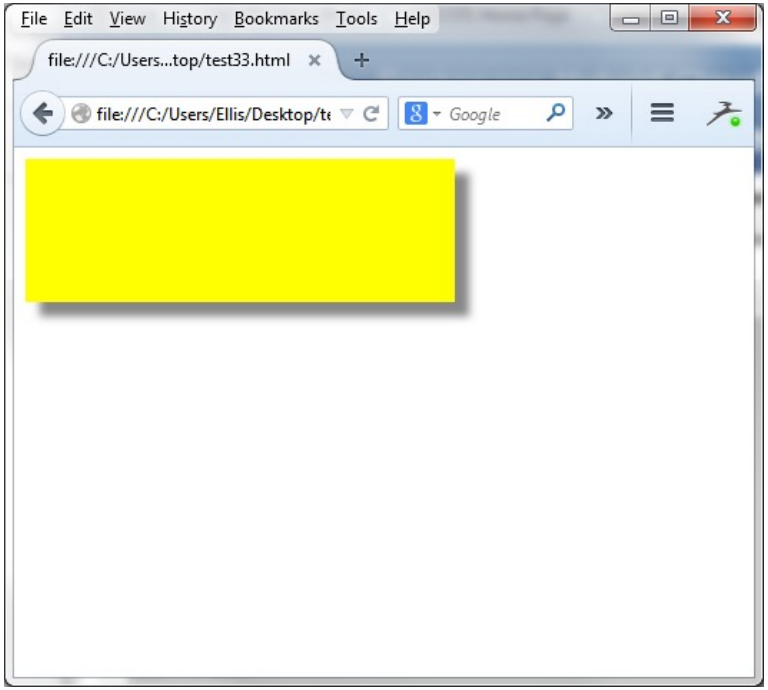
```
<style>
div {
    border: 15px solid transparent;
    width: 250px;
    padding: 10px 20px;  }
#round {
-webkit-border-image: url(border.png) 30 30 round; /* Safari */
    -o-border-image: url(border.png) 30 30 round; /* Opera */
    border-image: url(border.png) 30 30 round;   }
#stretch {
-webkit-border-image: url(border.png) 30 30 stretch; /* Safari */
    -o-border-image: url(border.png) 30 30 stretch; /* Opera */
    border-image: url(border.png) 30 30 stretch;   }
</style></head><body>
<p><b>Note:</b> Internet Explorer does not support the
border-image property.</p>
<p>The border-image property specifies an image to be used
as a border.</p>
<div id="round">Here, the image is tiled (repeated) to fill
'the area.</div><br>
<div id="stretch">Here, the image is stretched to fill the area.
</div>
<p>Here is the image used:</p>
<img src="border.png"></body></html>
```



Prefix no longer needed: https://www.w3schools.com/cssref/css3_pr_border-image.asp

Box Shadow

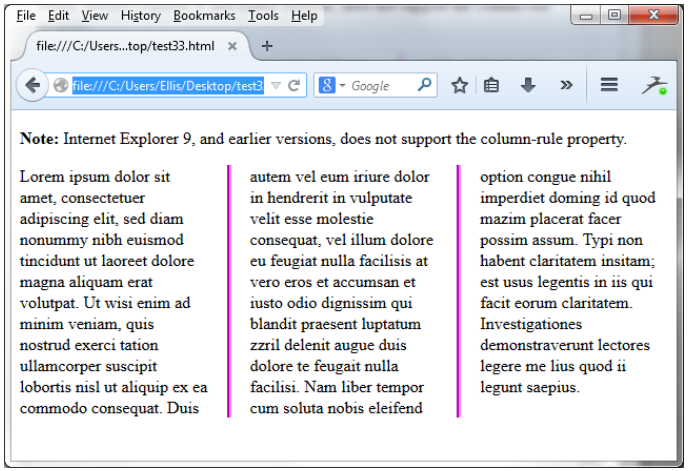
```
<style>
div {
    width: 300px;
    height: 100px;
    background-color: yellow;
    box-shadow: 10px 10px 5px #888888;
}
</style></head>
<body>
<div></div>
</body>
```



Multi-Column Layout

```
<!DOCTYPE html><html><head><style>
.newspaper {
-webkit-column-count: 3; /* Chrome, Safari, Opera */
-moz-column-count: 3; /* Firefox */
    column-count: 3;
-webkit-column-gap: 40px; /* Chrome, Safari, Opera */
-moz-column-gap: 40px; /* Firefox */
    column-gap: 40px;
-webkit-column-rule: 4px outset #ff00ff; /* Chrome, Safari, Opera */
-moz-column-rule: 4px outset #ff00ff; /* Firefox */
    column-rule: 4px outset #ff00ff;  }
</style></head><body>
<p><b>Note:</b> Internet Explorer 9, and earlier versions, does not
support the column-rule property.</p>
<div class="newspaper">
Lorem ipsum dolor sit amet, consectetuer adipiscing elit,
sed diam nonummy nibh euismod tincidunt ut laoreet dolore
magna aliquam erat volutpat. Ut wisi enim ad minim veniam,
quis autem velit esse molestie consequat, vel illum dolore
eu feugiat nulla facilisis. . . .
</div></body></html>
```

Prefix no longer needed: https://www.w3schools.com/cssref/css3_pr_column-count.asp



Advanced CSS

- Try the samples at W3Schools CSS Advanced:
  https://www.w3schools.com/css/

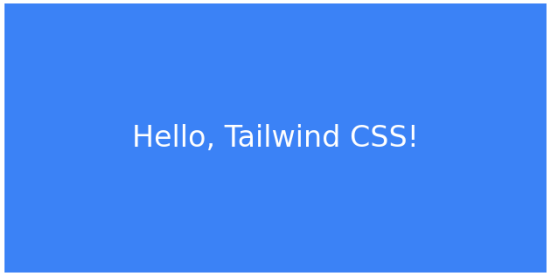
  - CSS Rounded Corners
  - CSS Border Images
  - CSS Backgrounds
  - CSS Gradients
  - CSS Shadows
  - CSS Transitions
  - CSS Animations
  - CSS Tooltips
  - CSS Flexbox

Tailwind CSS

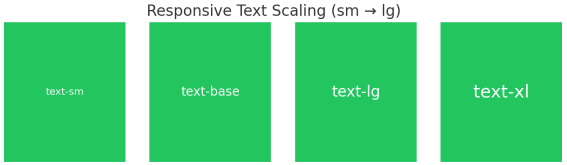
- Utility-First Framework
  - Provides low-level utility classes (e.g., p-4, text-center)
  - Enables building custom designs without leaving HTML
- Highly Customizable
  - Configurable design system via tailwind.config.js
  - Supports custom themes, colors, and responsive breakpoints
- Responsive & Mobile-First
  - Built-in responsive utilities (sm:, md:, lg:, xl:)
  - Encourages mobile-first development approach
- Performance-Oriented
  - Removes unused CSS in production builds (tree-shaking)
  - Keeps final CSS bundles small and efficient
- Used in many LLMs code generators
- See: https://tailwindcss.com/

Tailwind CSS - Examples

```
<div class="bg-blue-500 text-white p-4
rounded-lg shadow-md">
  Hello, Tailwind CSS!
</div>
```




```
<div class="p-4 bg-green-500 text-white
text-sm sm:text-base md:text-lg lg:text-xl
rounded">
  Responsive Text Example
</div>
```

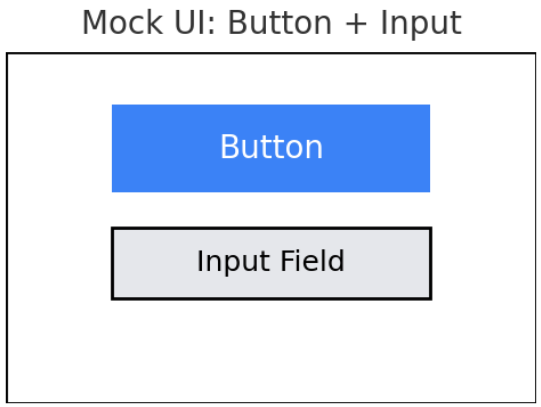


shadcn/ui Components

- Headless & Accessible
  - Provides un-styled, headless React components
  - Accessibility (ARIA) baked in by default
- Built on Tailwind CSS
  - Designed to pair seamlessly with Tailwind utilities
  - Ensures consistency with utility-first design
- Composable & Customizable
  - Small building blocks for UI (Button, Input, Dialog, etc.)
  - Easy to extend and theme to match brand identity
- Production-Ready
  - Includes motion/animation support
  - Optimized for modern frameworks like Next.js and Vite
- Used in many LLM code generators
- See: https://ui.shadcn.com/

shadcn/ui Components - Example

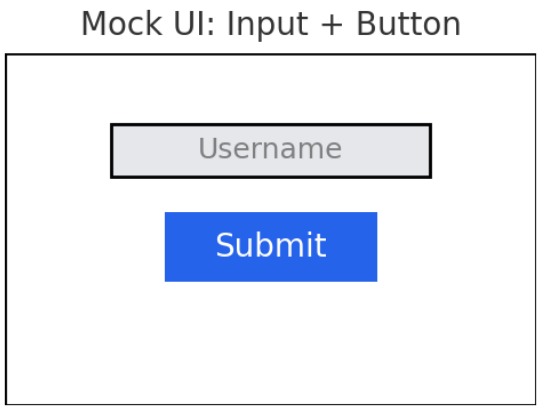
```
import { Button } from "@/components/ui/button"
import { Input } from "@/components/ui/input"

export default function Demo() {
  return (
    <div className="flex flex-col gap-2">
      <Button>Click Me</Button>
      <Input placeholder="Enter text..." />
    </div>
  )
}
```



shadcn/ui vs Tailwind-Only

- Tailwind-Only
  - Developers write utility classes directly in HTML/JSX
  - High flexibility, but repetitive for common UI patterns (e.g., buttons, forms)
- shadcn/ui
  - Provides pre-built, accessible React components
  - Reduces boilerplate by encapsulating common patterns (e.g., Button, Dialog)
- Benefit of Using Together
  - Tailwind handles fine-grained styling
  - shadcn/ui accelerates development with reusable building blocks

```
import { Button } from "@/components/ui/button"
import { Input } from "@/components/ui/input"

export default function Demo() {
  return (
    <div className="p-4 max-w-sm mx-auto shadow-lg rounded">
      <Input placeholder="Username" className="mb-2" />
      <Button className="bg-blue-600 hover:bg-blue-700">Submit</Button>
    </div>
  )
}
```

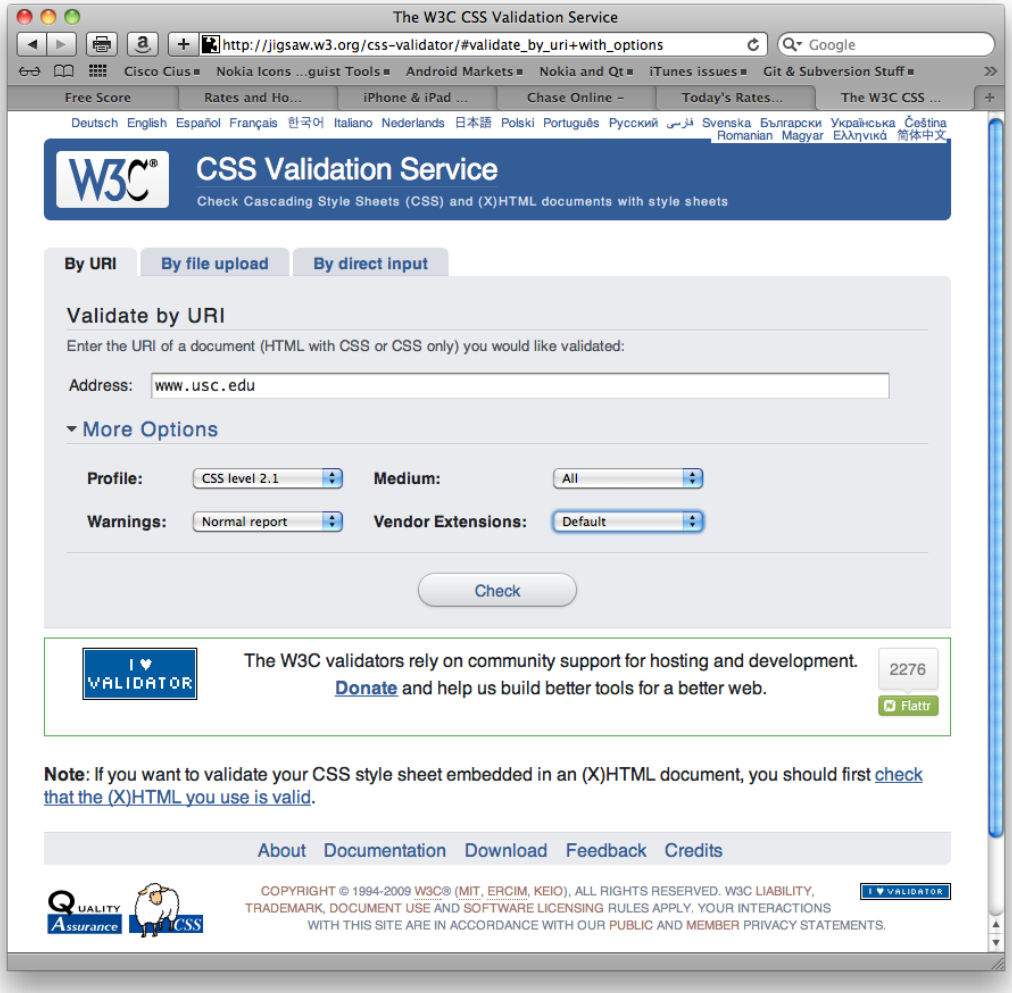


Discussion Section

Validations and Sample CSS

W3C CSS Validation Service

- Available at: http://jigsaw.w3.org/css-validator/

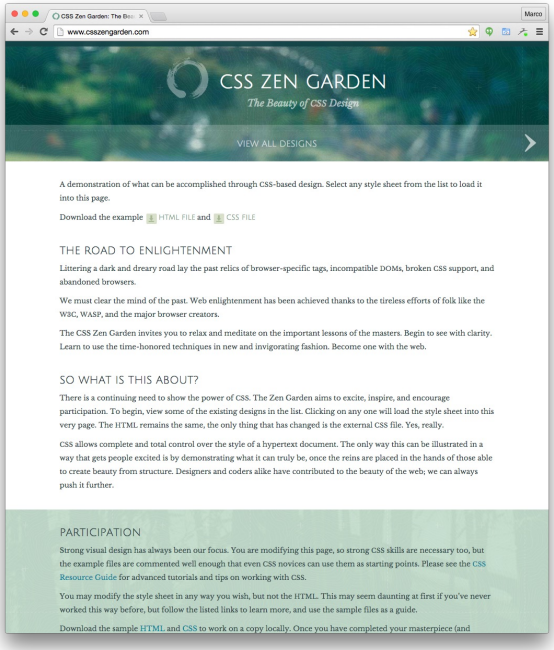


Discussion on Style

- For more examples of working with style sheets see

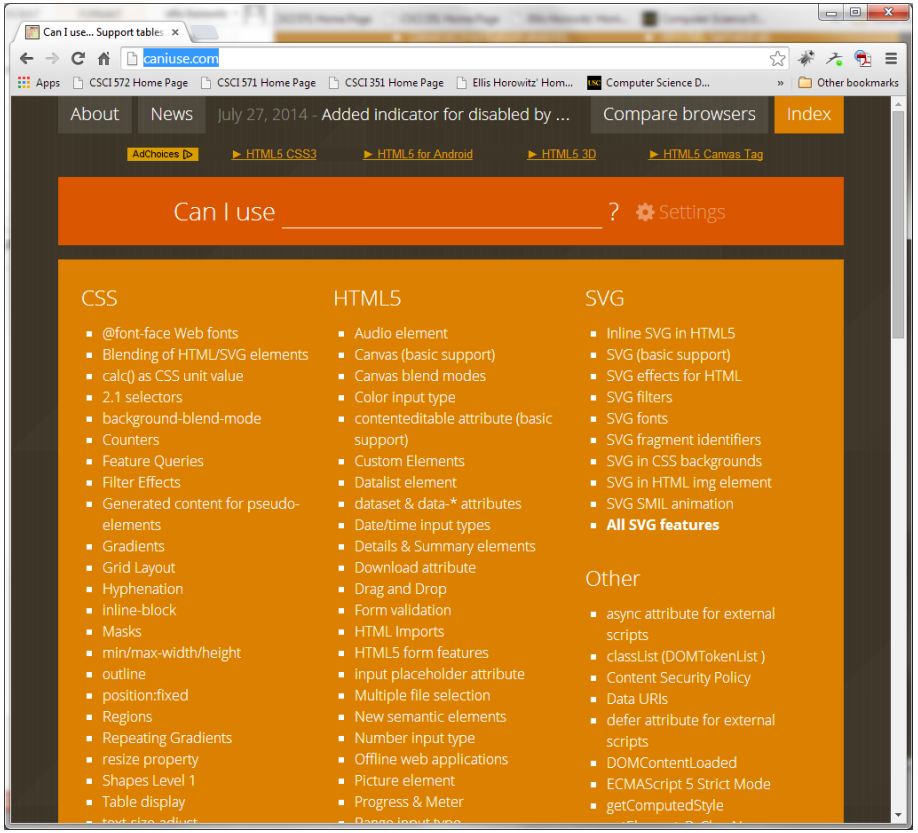
http://www.w3.org/Style/Examples/007/

http://www.csszengarden.com/



Here are eight versions of the same HTML page created using 8 different css files

CSS Implementations



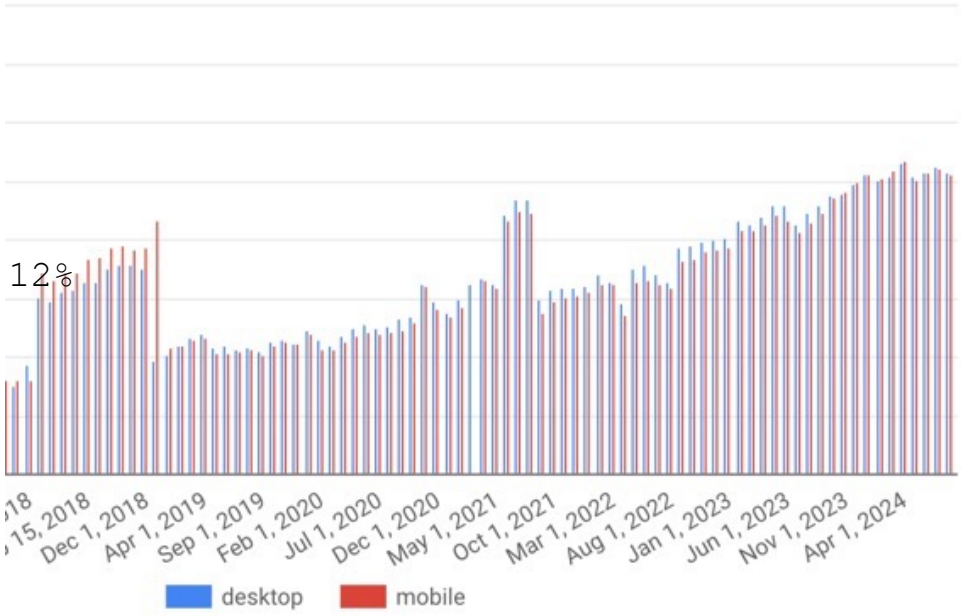
caniuse.com home page

CSS 2.1 selectors example

Stats

- Chrome anonymous stats (popular css properties):

  https://chromestatus.com/metrics/css/popularity



-webkit-line-clamp adoption

Property name	Percentage	
width	92.5422%	Timeline
display	92.0099%	Timeline
height	91.7091%	Timeline
padding	91.5247%	Timeline
position	91.1690%	Timeline
margin	91.1384%	Timeline
border	90.8517%	Timeline
top	89.9081%	Timeline
left	89.4888%	Timeline
color	88.9238%	Timeline
background-color	88.6703%	Timeline
margin-top	88.5238%	Timeline
font-size	88.3832%	Timeline
text-align	86.4264%	Timeline
font-weight	86.2063%	Timeline
overflow	86.0672%	Timeline
font-family	85.8612%	Timeline
background	85.8387%	Timeline
opacity	85.3521%	Timeline
box-sizing	84.5210%	Timeline
line-height	84.1971%	Timeline
border-radius	83.5621%	Timeline
z-index	83.3913%	Timeline
cursor	82.0438%	Timeline
text-decoration	81.6716%	Timeline
margin-left	81.4380%	Timeline
margin-bottom	81.1188%	Timeline
right	81.0583%	Timeline
margin-right	80.7627%	Timeline
max-width	80.0560%	Timeline
vertical-align	79.8206%	Timeline
transform	79.7625%	Timeline
bottom	79.0847%	Timeline
box-shadow	78.7405%	Timeline
padding-left	78.6932%	Timeline
white-space	78.4658%	Timeline
padding-top	78.2321%	Timeline

Browser updates

- Chrome:
  - Release: every 4 weeks major, bi-weekly minor
  - Roadmap: https://chromestatus.com/roadmap
  - EAP: Chrome Canary https://www.google.com/chrome/canary/ 
- Firefox:
  - Release: every 4 weeks (13 releases per year)
  - Roadmap: https://whattrainisitnow.com/release/?version=131
  - Train Calendar: https://whattrainisitnow.com/ Release notes: https://www.mozilla.org/en-US/firefox/releases/
  - EAP: Firefox Nightly 
- Safari:
  - Release: usually 4-6 times a year (no schedule)
  - EAP: Technology Preview https://developer.apple.com/safari/technology-preview/ 
  - WebKit Blog https://webkit.org/blog/

Codepen.io

- Codepen.io: “The best place to built, test and discover front-end code.”
- CSS Transform, animations on CODEPEN

  https://codepen.io/pork00chops/pen/vYXmbQW

  https://codepen.io/michalporag/pen/MWjRJBO
- SCSS on CODEPEN

  https://codepen.io/bennettfeely/pen/NWRmGYb
- More CODEPEN examples:

  https://www.google.com/search?q=codepen+examples