Quick Thought On Copyright

- Who ‘pushed the button’ with a motion sensor?

- (E.g. a security camera system that is triggered with motion)

Quick Aside on Macs

- GF has a Mac
- Went to apply for a job… and nothing. The website simply didn’t support mac at all, no browsers etc.

- The job listing didn’t even show up for mac users (but a direct link worked)

The joy of Programming

https://codepen.io/MartijnCuppens/pen/MXojmw

Identical code, 3 different ways to render the same thing.

Lectures 7 and Probably 8 or 9

Aesthetics and Cascading style sheets

Aesthetics

- It is the study of what is nice
- Applies to art, beauty, music, film, food etc.

- Designing things which are aesthetically pleasing is really quite challenging

Psychology of Aesthetics

- People react to aesthetic choices
- These reactions are real - some of them are universal, some are cultural
- Aesthetic choices can set a deliberate cultural tone

Key Concepts in Aesthetics

- Lines – e.g. things that should be read together should be on a line
- Shapes – elements that logically belong together should be contained in a shape (usually a rectangle/box)
- Colour palette – choice of colours
- Texture – the appearance of a material property can group elements together
- Form – 3d volume and shape of objects

Principles

- Unity – make stuff similar
- Space – you need some
- Hierarchy – significance between items (top is most important)
- Balance – perception of equal distribution, or unequal if that's what you want
- Contrast and scale – some things should stand out

Aesthetics are part of the Message

- Choosing a good font is quite difficult
- Choosing a good font is quite difficult
- Choosing a good font is quite difficult
- Choosing a good font is quite difficult
- Choosing a good font is quite difficult
- Choosing a good font is quite difficult
- Choosing a good font is quite difficult

Colour And Readability Matter

- Maybe you like Green text on a Black background

- What about blue?
- Maybe just a hint of orange that looks sort of pink (on one of my displays but not another)
- Maybe a brighter green would work?
- This isn’t a good choice

Aesthetics as separate from Function

- Things can be aesthetically pleasing for the sake of it (e.g. decorative)
- Something that looks horrible may never be able to succeed in the market no matter how technically capable
- Aesthetics can create a cycle of desirability – people like something and because it’s popular more people like it (e.g. white devices or metallic looks or leather seats in cars)

Aesthetics

Interesting, but not suitable for my living room, and I'd rather any building I might live in not have Something like this permanently

The Function of Aesthetics

- Aesthetics set the tone of your design
- Are you trying to set a product up as having a cultural theme?
- How is it going to be seen/heard/etc.? (a big black auditorium may warrant different style than a small well lit classroom)
- Why is it going to be seen/heard/etc.?

Aesthetics, Visual stuff

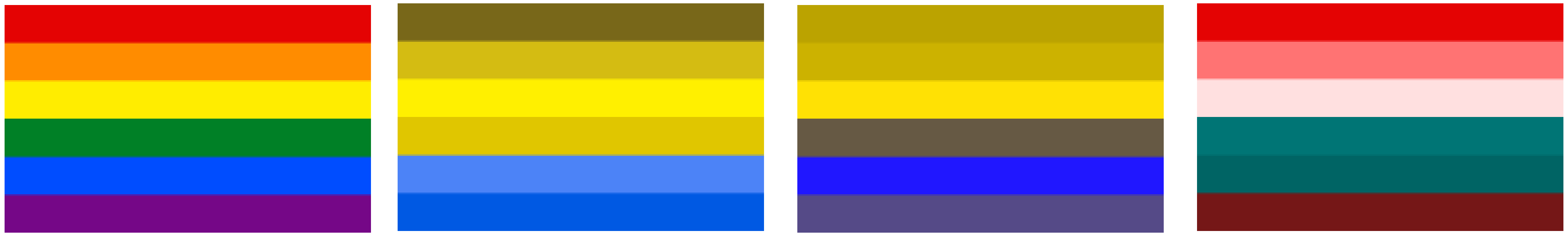
- We are going to start looking at some of the visual aspects of aesthetics

- When we get to game design more of this comes together as you start incorporating architecture, dress, culture etc.

Colours

- One could do the history of art and colour as an entire degree
- Colours have meaning
- That meaning sometimes changes over time
- Pink for girls blue for boys is only about a century old

Accessibility – Colour Vision



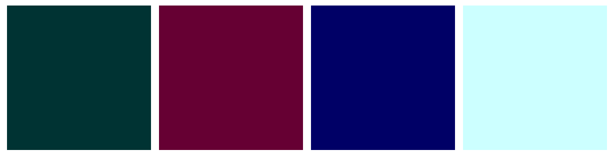
- Making things visible is definitely an accessibility problem
- But choosing the right colours is more than just about making them visible

Chrome Extension

- https://chrome.google.com/webstore/detail/lets-get-color-blind/bkdgdianpkfahpkmphgehigalpighjck

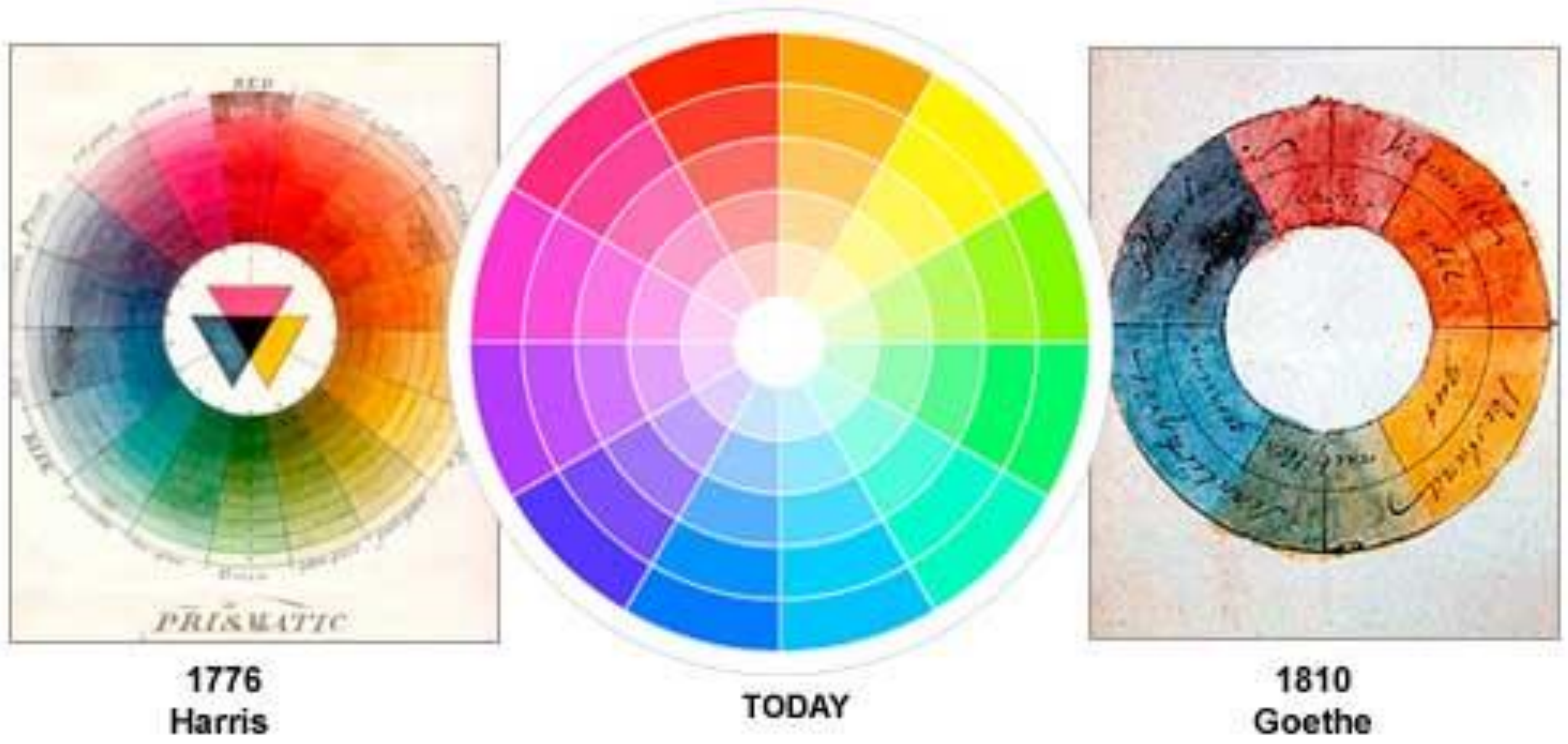
- There’s a firefox version as well. It’s called “Lets get color blind”

Colour- A technical nightmare



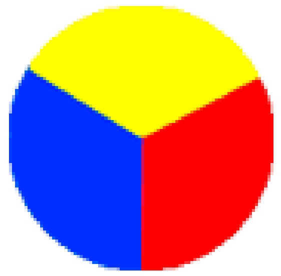
- They are supposed to be, very dark green, deep burgundy, medium/darkblue, and blue-green pastel
- Different monitors with display those better or worse, and gamma settings on your display may make them different
- You don’t need to make it work but understand that the same colour can look different in two places.

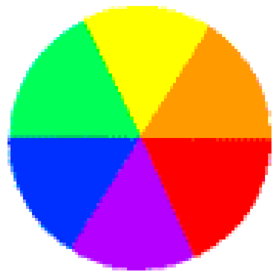
The Colour wheel

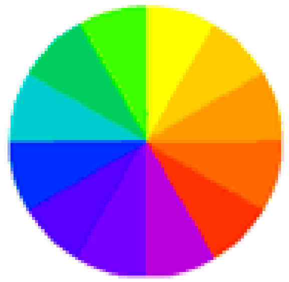


With some history on the sides

Primary Secondary Tertiary

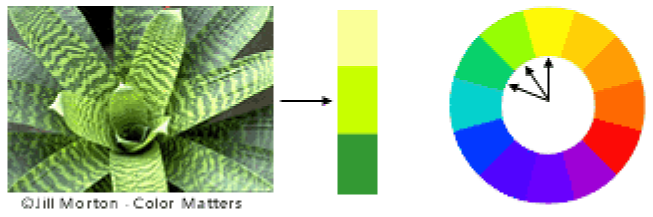

Primary Colors

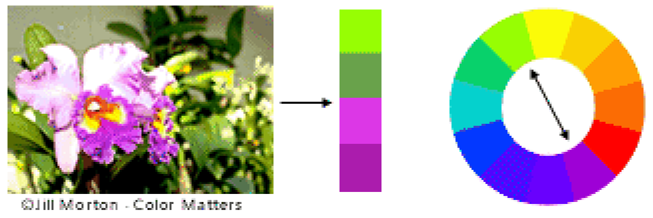

Secondary Colors


Tertiary Colors

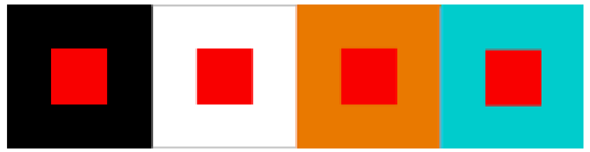
- You might have learned about this in art at school (I did)
- But it’s useful to reiterate terminology

- There are lots of theories on what makes for a good colour choice
- E.g. on the right, analogous (top) or complimentary (bottom)





Colour Contrasts



- Contrast

- Different options have different choices for readability

Warm colours

- Passion, happiness, enthusiasm
- Red is tricky…. Violence and war and death, and love and fertility. Because those obviously go together /sarcasm
- In china red is prosperity and happiness
- Also, Communism, or the british empire.

Yellow

- yellow has very different connotations. In Egypt, for example, yellow is for mourning. In Japan, it represents courage, and in India it’s a colour for merchants
- Quote from source in PPTX note
- Supposedly happy in the west. Also gender neutral for babies

Cool colours

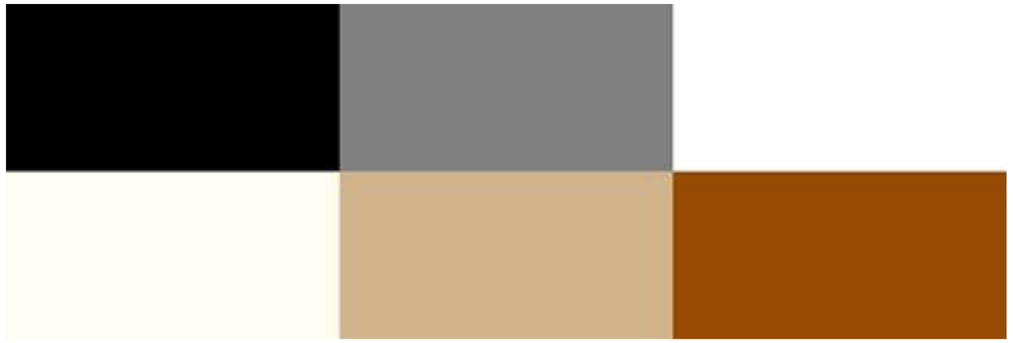
- Green is strongly associated with plants and earthy stuff
- Blue and green are both calming (that doesn't seem to work on facebook)
- Dark blue is popular for corporations because it's strong and reliable… apparently

Purple (violet)

- Used to be very royal. It’s a nightmare to make purple dye in Europe*, so only the rich had it
- Colour of mourning for widows in Thailand.

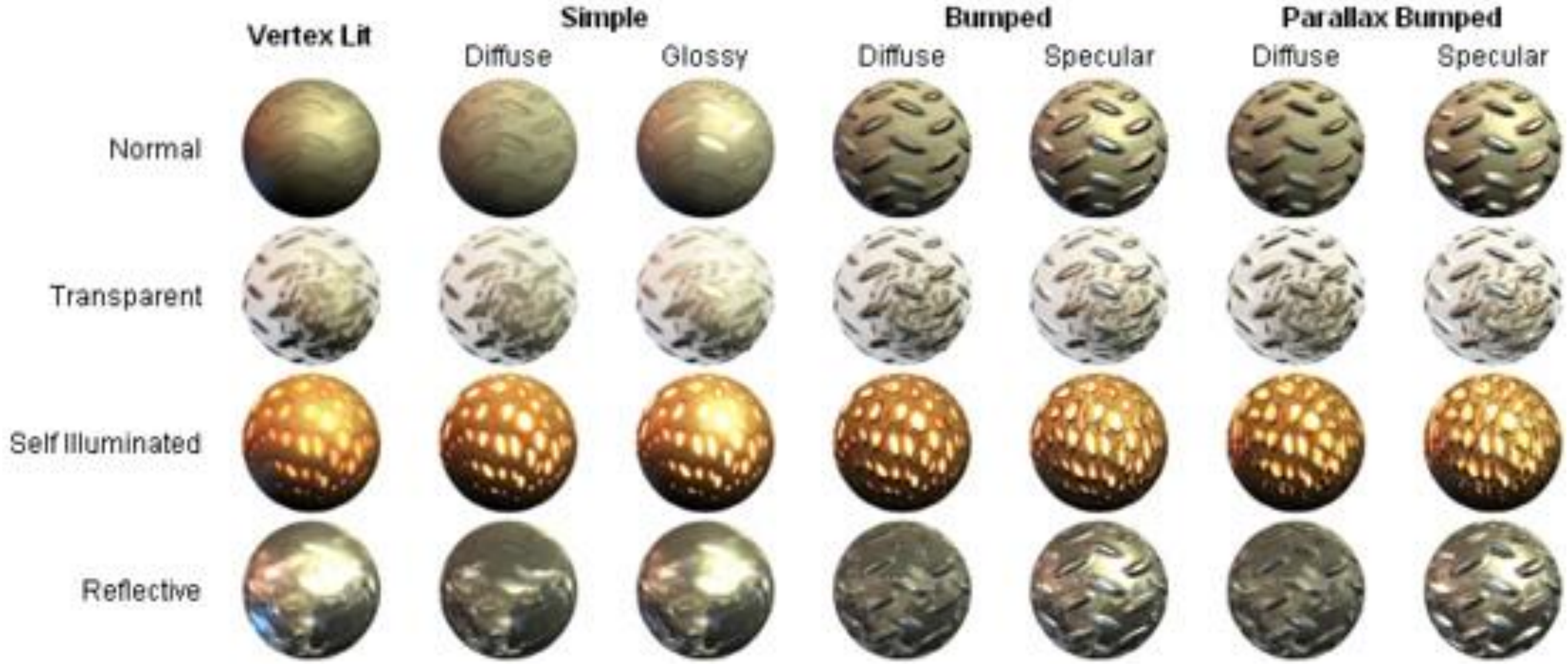
- *Carrots are naturally purple but don’t make purple dye. Purple dye in Europe is from thousands of crushed sea snails that are processed somehow

Neutrality



- Convenient backdrops in design, where you accent them with another colour
- Brown is associated with wood
- Black is very strong (sometimes attached to things like death too, which can be bad)
- Offwhites and ivory can be difficult to make visible

Textures and Materials



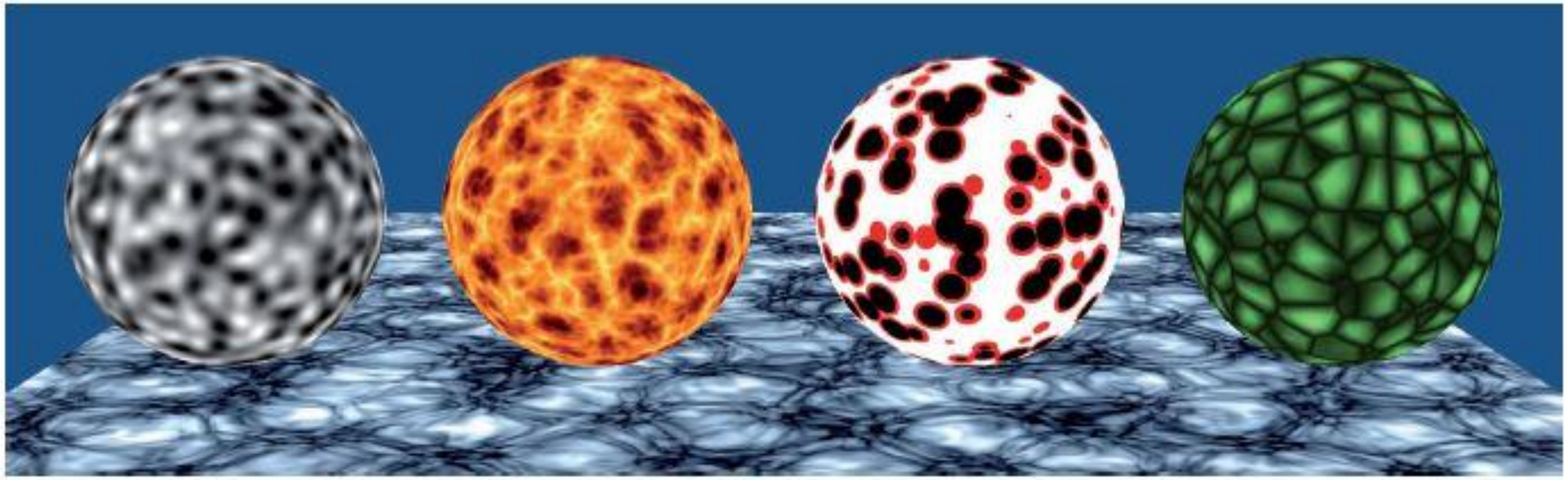
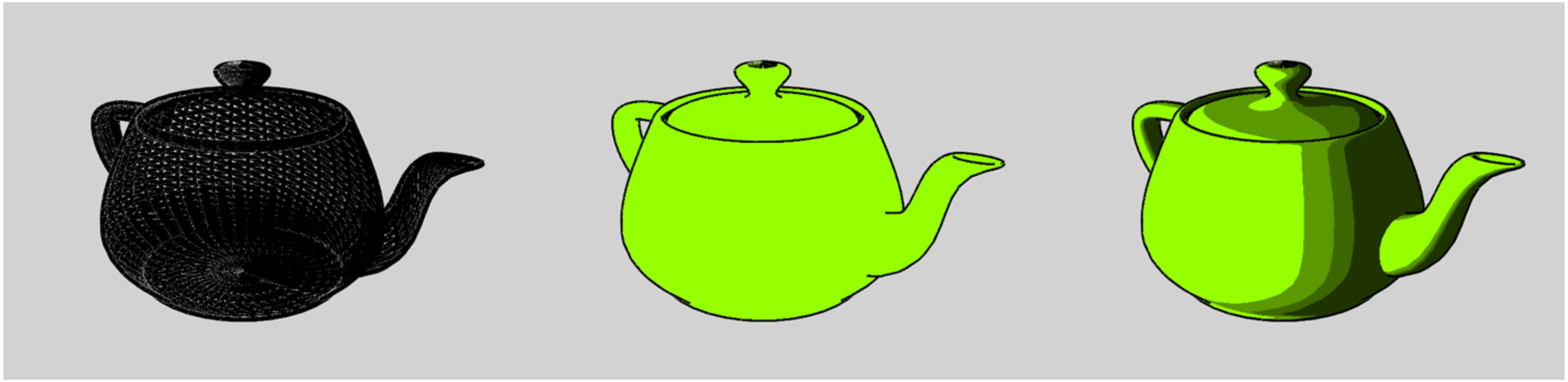


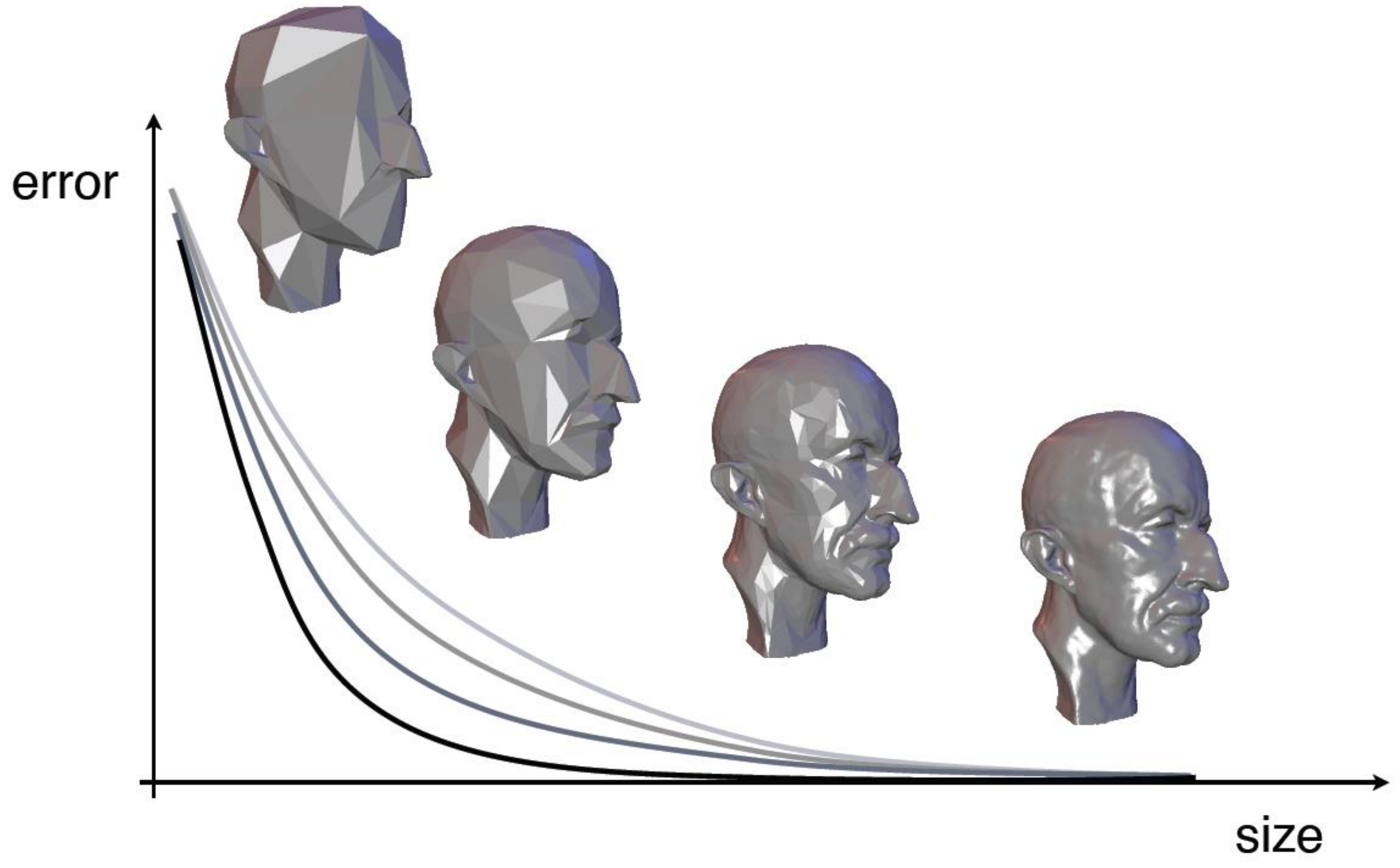
Figure 7.1. Examples of procedural textures. A modern GPU renders this image at full screen resolution in a few milliseconds.

This was about the pinnacle of real time computer generated materials and visual effects ~10 years
Fortunately, tech has moved on.

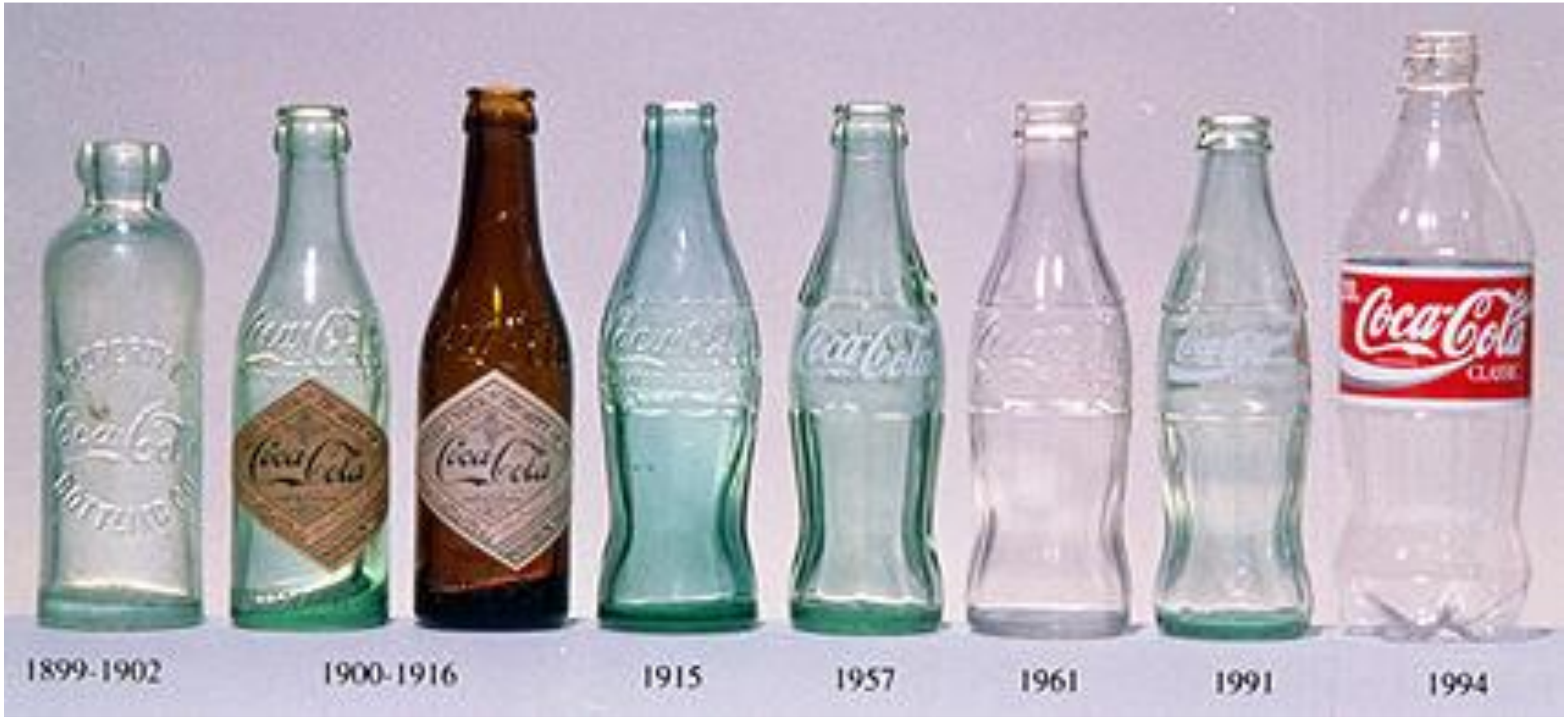
Textures and other effects



Size-Quality Tradeoff



Shape



1899-1902
1900-1916
1915
1957
1961
1991
1994
Coca-Cola
Classic

Coca-Cola
Coca-Cola
Coca-Cola
Coca-Cola
Coca-Cola
Coca-Cola
Coca-Cola

Iconic Style

- Shapes are one thing
- What about architecture?

Visual Themes

- The idea is that we take what we know about existing real world architecture, and use it to build visual themes (for fictional worlds) in game/level/movies/fiction

E.G. Jedi

- Shaolin monks
- Japanese Samurai
- Knights Templar (celibate military order vows of poverty etc.)

Choosing an aesthetic style

- When you choose and aesthetic style you are making a choice about cultural impact

Kawaii

Cute in Japan

- Grew out of the anime and Manga industry
- It’s a bit like 40 year old men watching my little pony

- Has major companies putting out cute mascots
- Women who feign particular attributes to be more attractive

Cute in Japan

- Has spread into fashion and other media
- As we saw previously, construction markers on roads that look like pink rabbits I think it was

Cultural Norms

- Cuteness as an aesthetic style is a cultural quirk of Japan that is starting to spread
- But of course we influence them too

Digital Aesthetics

- Of course really what we’re interested in are digital aesthetics.
- What should a website look like? Naturally there’s no right answer, but are want to apply principles to make that happen.

Aesthetics on the web

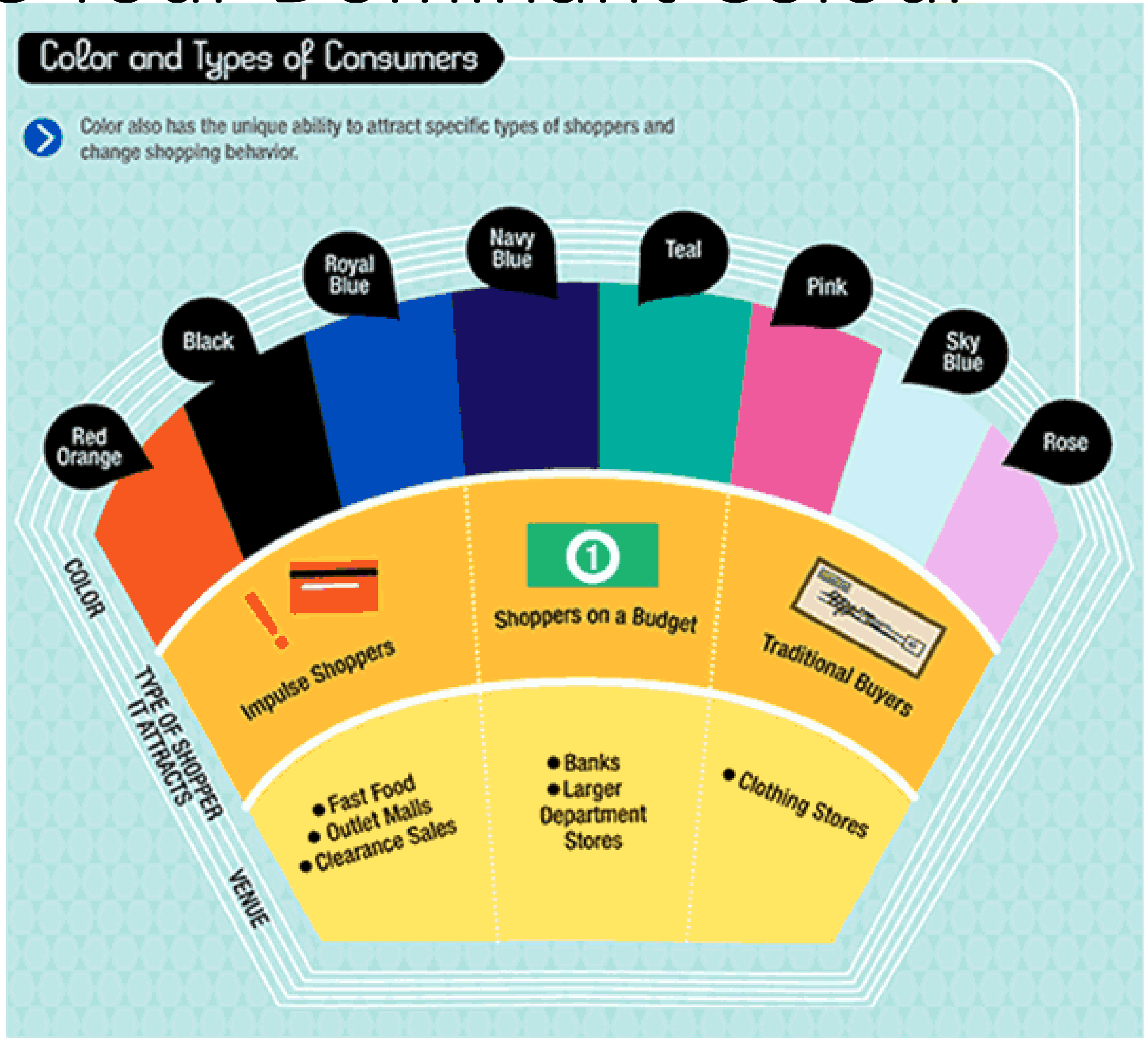
- Colour
- Images
- Text
- Navigation
- Alignment
- Consistency and Continuity
- White Space

A good read

- https://www.websitebuilderexpert.com/designing-websites/how-to-choose-color-for-your-website/

- Has a good overview of how to do all of this (better than Sri could come up with on his own anyway)

Choose Your Dominant Colour



PERSONALITY of COLOR

What color should you choose for your website?

GREEN
Represents wealth, health, tranquility, and nature.

The easiest color for the eye to process, so it has relaxation effects.
Green is the No.2 most preferred color by both men and women.

bp · WHOLE FOODS MARKET · ANIMAL PLANET · Tropicana

YELLOW
Represents youthfulness, optimism & cheerfulness.

Often used to grab the attention of the audience.
Yellow can put strain on the eyes, so you want to use it sparingly.

Hertz · NATIONAL GEOGRAPHIC · BEST BUY

ORANGE
Represents friendliness, enthusiasm & creativity.

Promotes people to take action: Buy & Subscribe.
Orange attracts impulse shoppers.

amazon · Payless · HARLEY-DAVIDSON · Crush

RED
Represents passion, energy, urgency, excitement, vibrancy & danger.

Often used to create urgency for people to buy.
Effective in triggering strong emotional reactions.
Restaurants use it to stimulate appetite.

TARGET · kmart · Heinz · RED ROBIN · H&M · Coca-Cola

PINK
Represents feminine, sweetness, innocence, fertility & romance.

Often used to market services and products to women and young girls.

PINK VICTORIA'S SECRET · Barbie · lyft · T

PURPLE
Represents royalty, wealth, success & wisdom.

Often used in beauty or anti-aging products.
Purple has a soothing and calming effect on people.

Crown Royal · QATAR AIRWAYS · THAI · Cadbury

BLUE
Represents trust, security, stability, peace & calmness.

Often used in businesses and banks to create sense of security & trust in the brand.
Blue is the No.1 preferred color by both men & women .

Oral-B · AMERICAN EXPRESS · GE · DELL

GRAY
Represents neutral, simplicity, calm, futuristic & logic.

It lacks emotion and is associated with technology, industry, precision, control, competence and even sophistication.

BOSCH · TESLA

BLACK
Represents power, luxury, sophistication & elegance.

Often used to market luxury brands to evoke professionalism, strength & precision.

CHANEL · ROLLS ROYCE · UNDER ARMOUR · L'ORÉAL PARIS

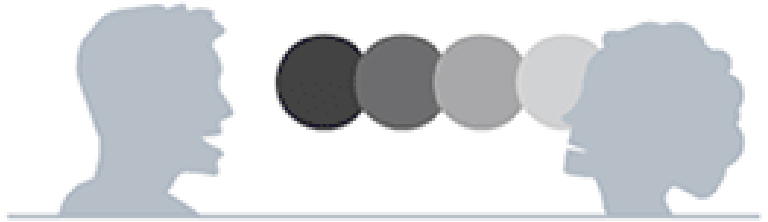
Bright vs. Soft Colors

In the experiment, both men and women had the same general preference when it came to light and dark colors. However, the experiment showed that women gravitate toward soft colors, while men like bright ones.

Men prefer **bright colors**
Women prefer **soft colors**

Achromatic

As a general rule, men tolerate achromatic colors more than women. Achromatic colors are those which have no hue—like black, white and shades of gray.



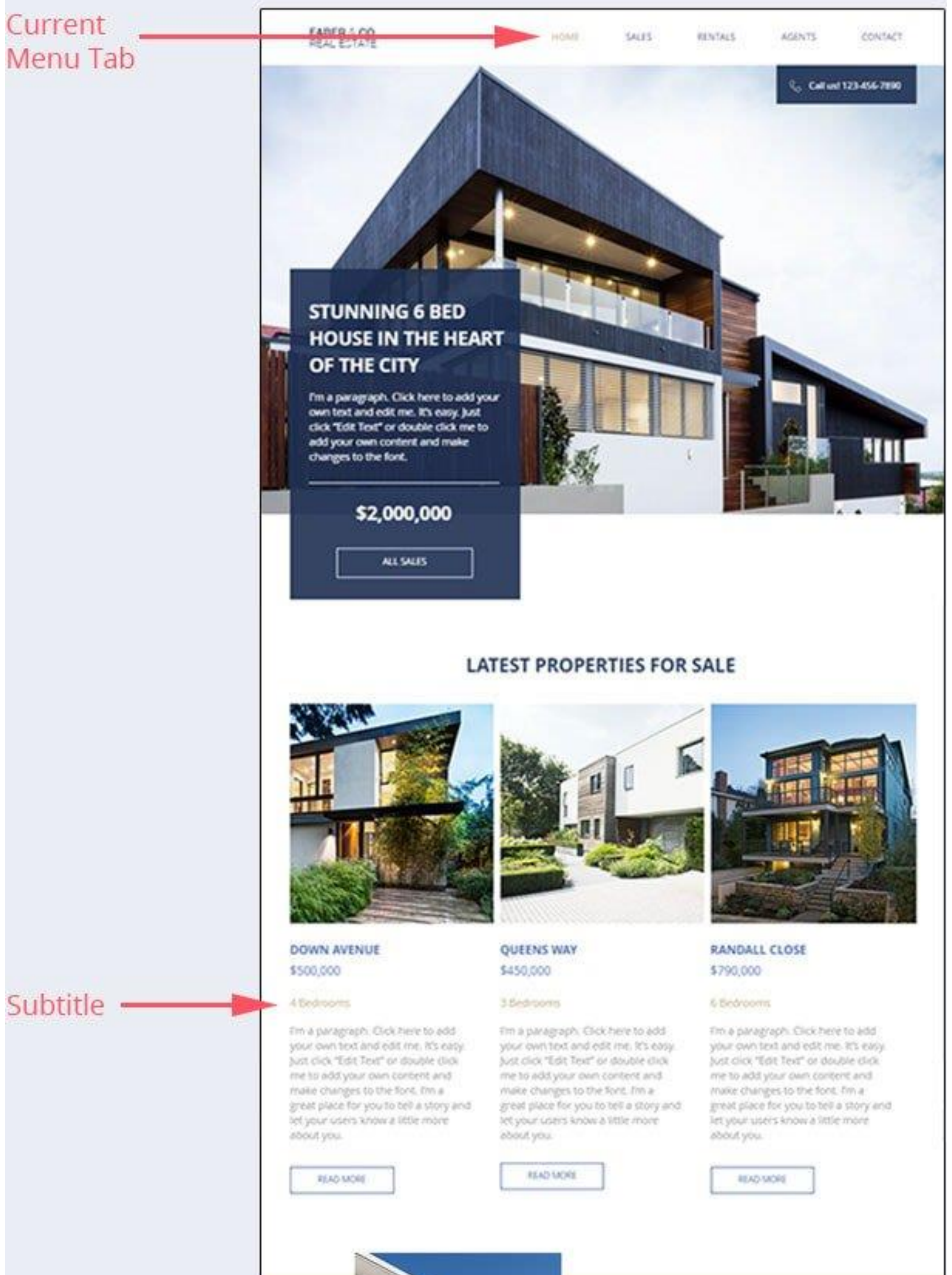
Tints vs. Shades

A McInnis and Shearer experiment found that women preferred tints more than shades. It's often proposed that the reason is due to their higher consciousness of specific colors. **A "tint" is simply any color with white added.** A color scheme using tints is soft, youthful and soothing. **A "shade" is simply any color with black added.** Shades are deep, powerful and mysterious.

Shades
Tints

TARGETING WOMEN	✓ *Colors women love the most*[8]	BLUE	PURPLE	GREEN
	✗ *Colors women hate the most*[8]	ORANGE	BROWN	GRAY
TARGETING MEN	✓ *Colors men love the most*[8]	BLUE	GREEN	BLACK
	✗ *Colors men hate the most*[8]	BROWN	ORANGE	PURPLE

Where to use accent color in your website



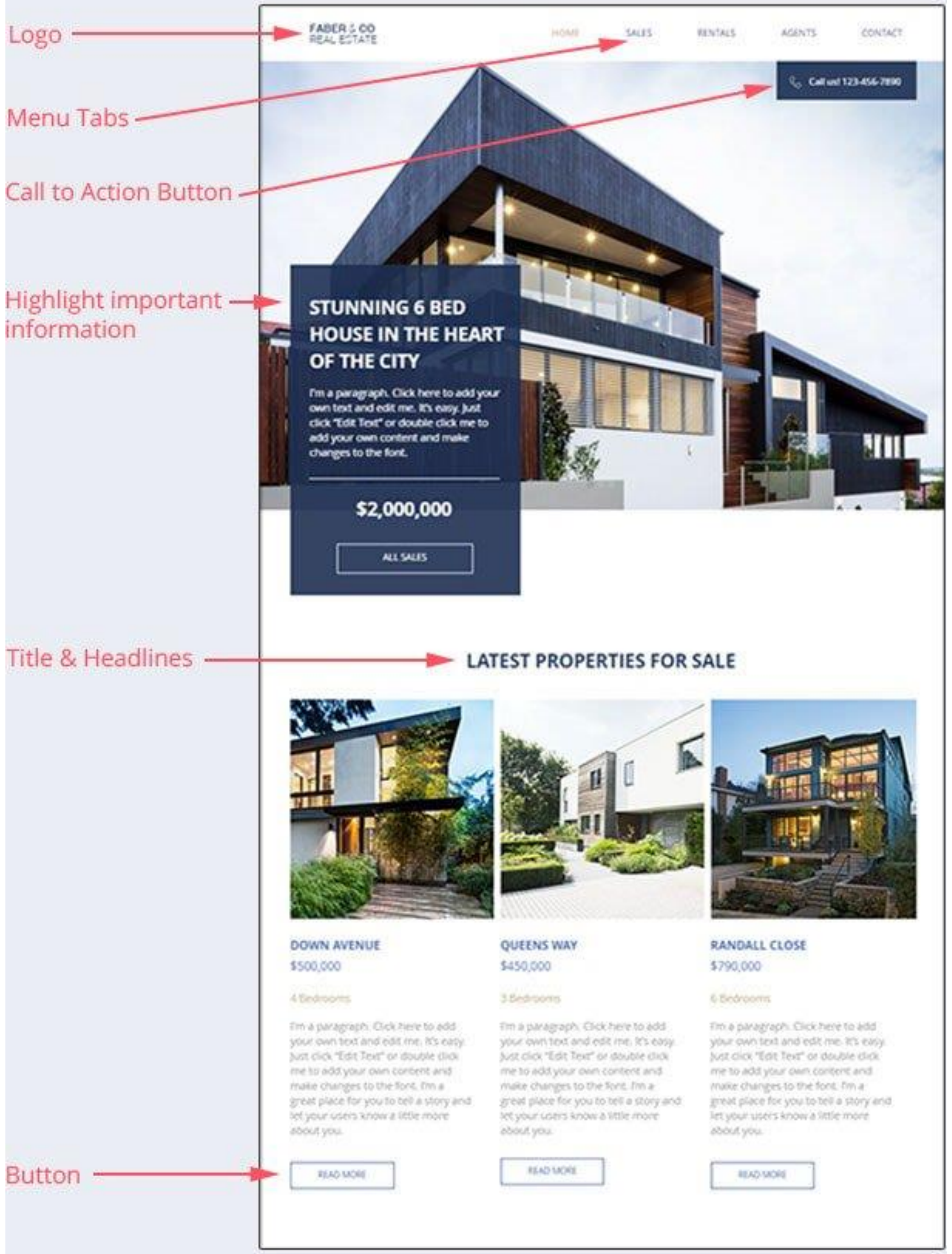
STUNNING 6 BED HOUSE IN THE HEART OF THE CITY

I'm a paragraph. Click here to add your own text and edit me. It's easy. Just click "Edit Text" or double click me to add your own content and make changes to the font.

LATEST PROPERTIES FOR SALE

I'm a paragraph. Click here to add your own text and edit me. It's easy. Just click "Edit Text" or double click me to add your own content and make changes to the font. I'm a great place for you to tell a story and let your users know a little more about you.

Where to use dominant color in your website



STUNNING 6 BED HOUSE IN THE HEART OF THE CITY

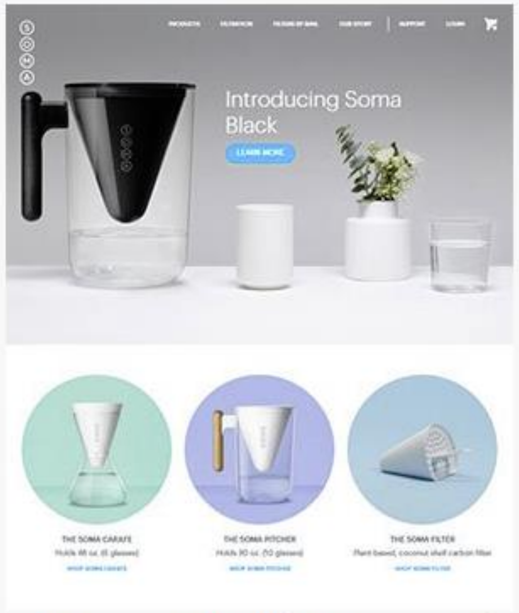
I'm a paragraph. Click here to add your own text and edit me. It's easy. Just click "Edit Text" or double click me to add your own content and make changes to the font.

LATEST PROPERTIES FOR SALE

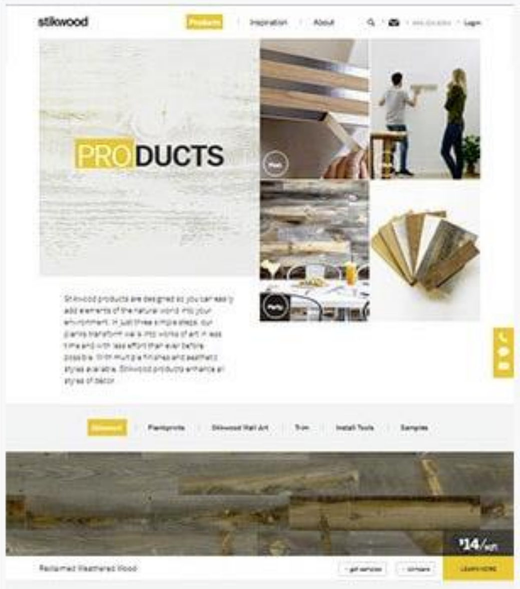
I'm a paragraph. Click here to add your own text and edit me. It's easy. Just click "Edit Text" or double click me to add your own content and make changes to the font. I'm a great place for you to tell a story and let your users know a little more about you.

Background Colour?

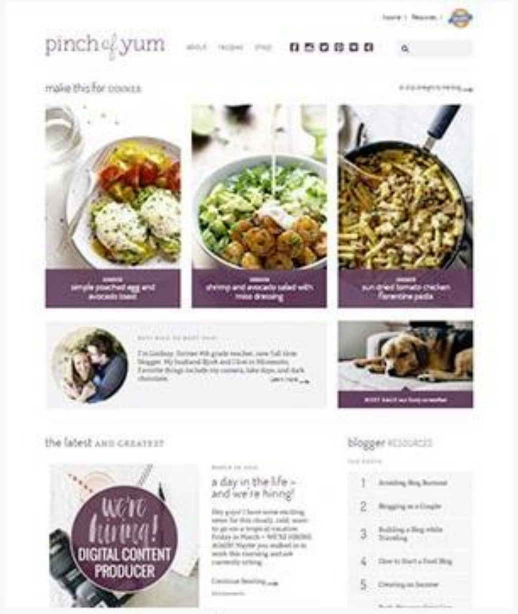
- Selling stuff? Use white



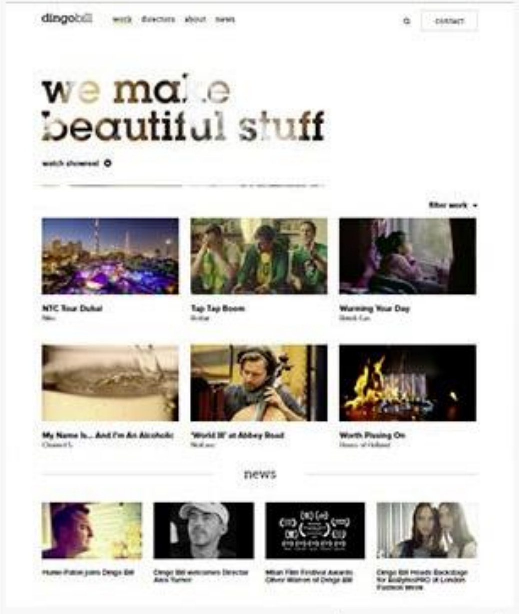
eCommerce Website (drinksoma.com)



eCommerce Website (stikwood.com)

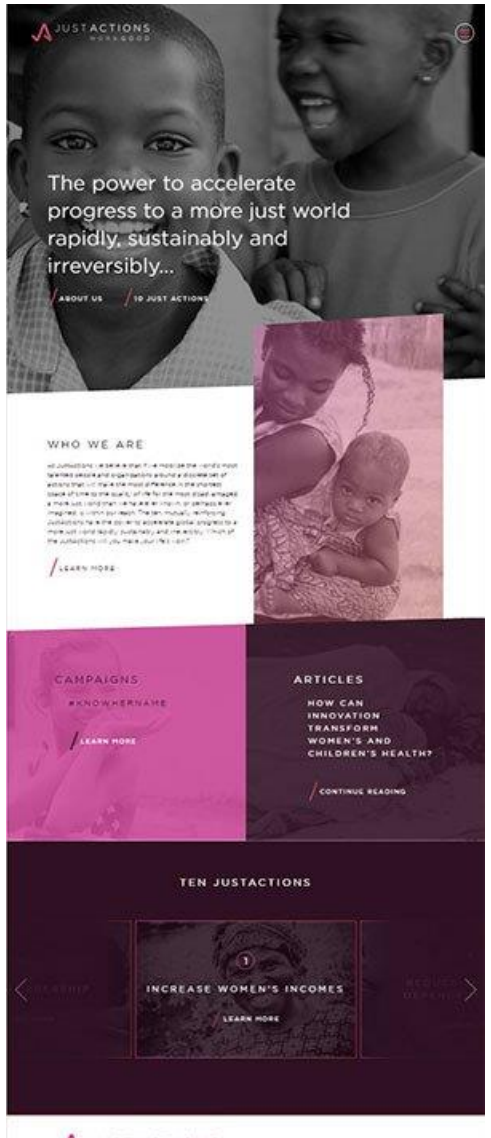


Food Blog (pinchofyum.com)



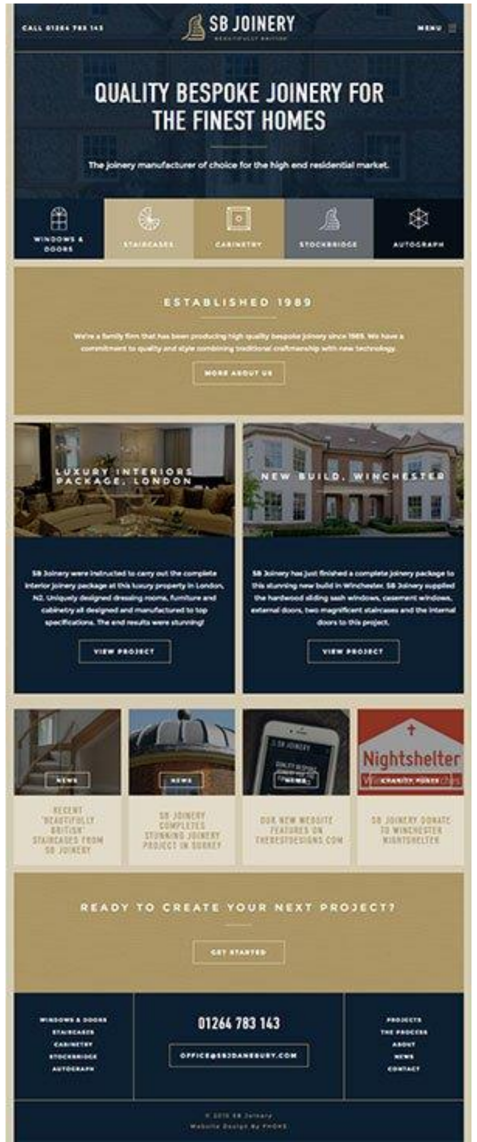
Film Production Company (dingobill.com)

Promoting a Brand? Use Colour as a BG



The power to accelerate progress to a more just world rapidly, sustainably and irreversibly...

HOW CAN INNOVATION TRANSFORM WOMEN'S AND CHILDREN'S HEALTH?



QUALITY BESPOKE JOINERY FOR THE FINEST HOMES

The joinery manufacturer of choice for the high end residential market.

To do that

- To make website look a particular way – buttons have a particular form, links colour a certain way etc. We use something called Cascading style sheets…

11

INTRODUCING CSS

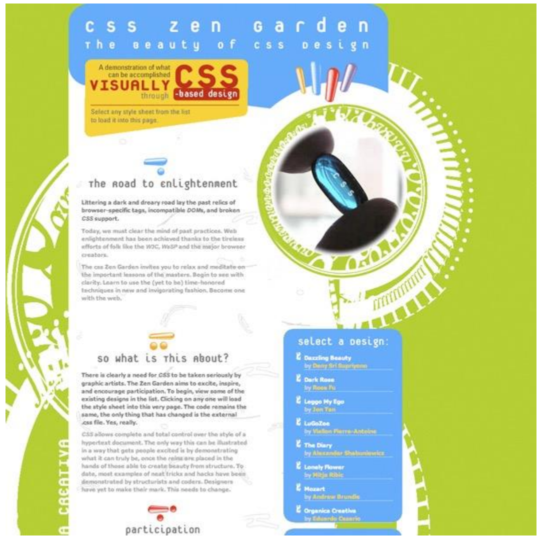
- The benefits of CSS
- Understanding document structure
- Writing style rules
- Attaching styles to the HTML document
- Inheritance
- The cascade
- The box model
- CSS units of measurement

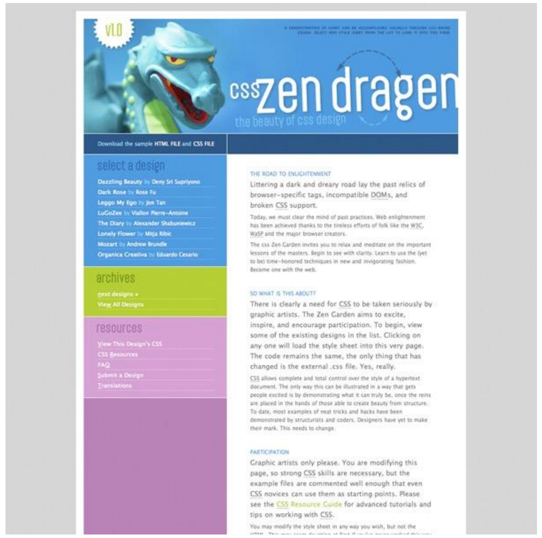
The Benefits of CSS

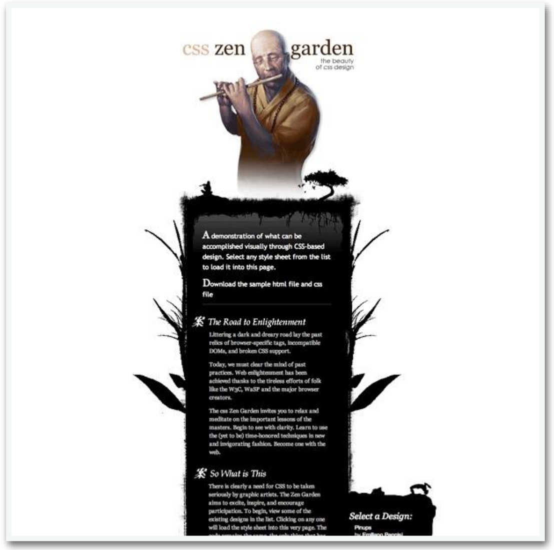
- Precise type and layout control
- Less work: Change look of the whole site with one edit
- Accessibility: Markup stays semantic
- Flexibility: The same HTML markup can be made to appear in dramatically different ways

Style Separate from Structure

- These pages have the exact same HTML source but different style sheets:


css zen garden
the beauty of css design
A demonstration of what can be accomplished VISUALLY through CSS-based design
The Road to Enlightenment
So What is This About?
participation
select a design:





(csszengarden.com)

How Style Sheets Work

1. Start with a marked up document (like HTML, but could be another XML markup language).
2. Write styles for how you want elements to look using CSS syntax.
3. Attach the styles to the document (there are a number of ways).
4. The browser uses your instructions when rendering the elements.

Style Rules

Each rule *selects* an element and *declares* how it should display.

```
h1 { color: green; }
```

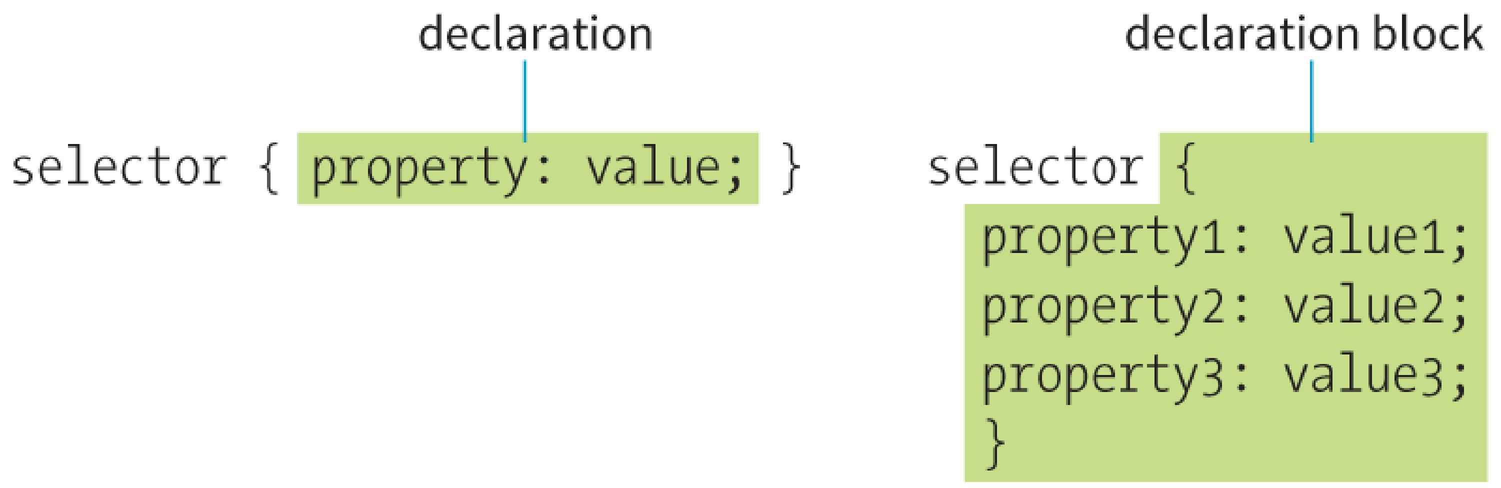
This rule selects all `h1` elements and declares that they should be green.

```
strong { color: red; font-style: italic; }
```

This rule selects all `strong` inline elements and declares that they should be red and in an italic font.

Style Rule Structure

- A style rule is made up of a **selector** a **declaration**.
- The declaration is one or more **property / value** pairs.



```
selector { property: value; }
```

```
selector {
  property1: value1;
  property2: value2;
  property3: value3;
  }
```

Selectors

There are many types of selectors. Here are just two examples:

```
p {property: value;}
```

Element type selector: Selects all elements of this type (**p**) in the document.

```
#intro {property: value}
```

ID selector (indicated by the # symbol) selects by ID value. In the example, an element with an id of “intro” would be selected.

Declarations

The **declaration** is made up of a **property/value** pair contained in curly brackets { }:

```
selector { property: value; }
```

Example

```
h2 { color: red;
     font-size: 2em;
     margin-left: 30px;
     opacity: .5;
   }
```

Declarations (cont’d)

- End each declaration with a semicolon to keep it separate from the next declaration.
- White space is ignored, so you can stack declarations to make them easier to read.
- Properties are defined in the CSS specifications.
- Values are dependent on the type of property:
  - Measurements
  - Keywords
  - Color values
  - More

CSS Comments

```
/* comment goes here */
```

- Content between /* and */ will be ignored by the browser.
- Useful for leaving notes or section labels in the style sheet.
- Can be used within rules to temporarily hide style declarations in the design process.

Adding Styles to the Document

There are three ways to attach a style sheet to a document:

External style sheets

A separate, text-only *.css* file associated with the document with the **`link`** element or **`@import`** rule

Embedded style sheets

Styles are listed in the **`head`** of the HTML document in the **`style`** element.

Inline styles

Properties and values are added to an individual element with the **`style`** attribute.

External Style Sheets

The style rules are saved in a separate text-only *.css* file and attached via **`link`** or **`@import`**.

Via **`link`** element in HTML:

```
<head>
  <title>Titles are require</title>
  <link rel="stylesheet" href="/path/example.css">
</head>
```

Via **`@import`** rule in a style sheet:

```
<head>
  <title>Titles are required</title>
  <style>
    @import url("/path/example.css");
    p {font-face: Verdana;}
  </style>
</head>
```

Embedded Style Sheets

Embedded style sheets are placed in the `head` of the document via the `style` element:

```
<head>
  <title>Titles are required</title>
  <style>
    /* style rules go here */
  </style>
</head>
```

Inline Styles

Apply a style declaration to a single element with the **`style`** *attribute*:

```
<p style="font-size:
large;">Paragraph text...</p>
```

To add multiple properties, separate them with semicolons:

```
<h3 style="color: red; margin-top: 30px;">Intro</h3>
```

Document Structure

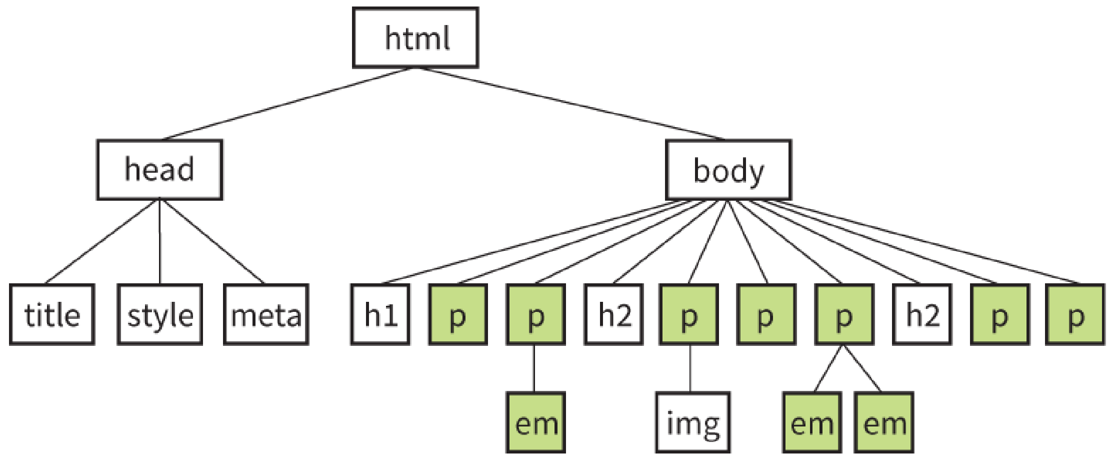
Documents have an implicit structure.

We give certain relationships names, as if they're a family:

- All the elements contained in a given element are its **descendents.**
- An element that is directly contained within another element is the **child** of that element.
- The containing element is the **parent** of the contained element.
- Two elements with the same parent are **siblings.**

Inheritance

- Many properties applied to elements are passed down to the elements they contain. This is called **inheritance**.
- For example, applying a sans-serif font to a **`p`** element causes the **`em`** element it contains to be sans-serif as well:



```
p {font-size: large; font-family: sans-serif;}
```

Inheritance (cont’d)

- Some properties inherit; others do not. Properties related to text usually inherit; properties related to layout generally don’t.
- Styles explicitly applied to specific elements override inherited styles.
- You’ll learn to use inheritance strategically to keep your style rules simple.

The Cascade

- The **cascade** refers to the system for resolving conflicts when several styles apply to the same element.
- Style information is passed down (it “cascades” down) until overwritten by a style rule with more **weight**.
- Weight is considered based on:
    - Priority of style rule source
    - Specificity of the selector
    - Rule order

The Cascade: Priority

Style rules from sources higher in this list override rules from sources listed below them.

- Any style marked as `!important` by the user (to accommodate potential accessibility settings)
- Any style marked `!important` by the author (of the web page)
- Author styles (style sheets created in web site production)
- User styles (added by the reader)
- User agent styles (browser defaults)

The Cascade: Specificity

- When two rules in a single style sheet conflict, the type of selector is used to determine which rule has more weight.
- For example, ID selectors are more specific than general element selectors.

NOTE: Specificity will be discussed once we have covered more selector types.

The Cascade: Rule Order

- When two rules have equal weight, rule order is used. Whichever rule appears last "wins."

```
<style>
  p {color: red;}
  p {color: blue;}
  p {color: green;}
</style>
```

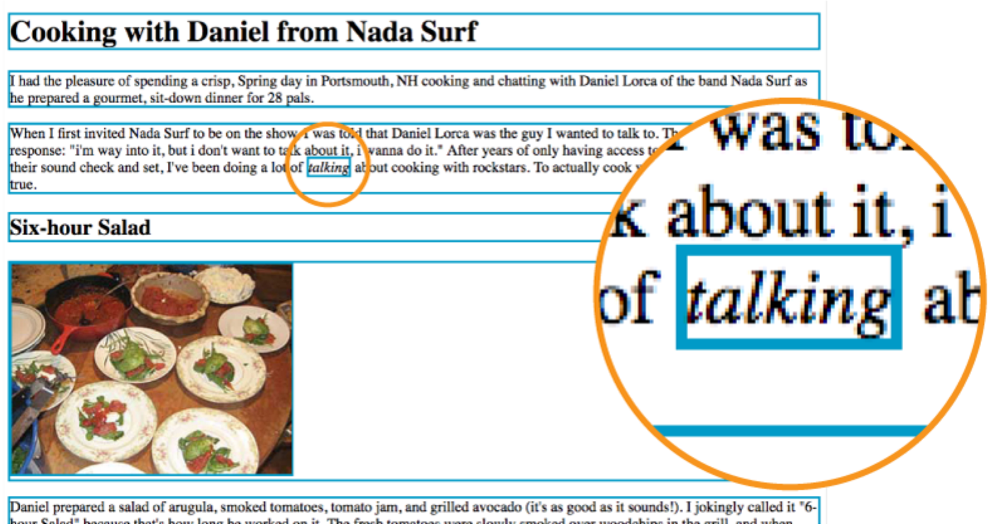
In this example, paragraphs would be green.

- Styles may come in from external style sheets, embedded style rules, and inline styles. The style rule that gets parsed last (the one closest to the content) will apply.

The Box Model

Browsers see every element on the page as being contained in a little rectangular box. Block elements and inline elements participate in the box model.

In this example, a blue border is added to all elements.



Cooking with Daniel from Nada Surf

I had the pleasure of spending a crisp, Spring day in Portsmouth, NH cooking and chatting with Daniel Lorca of the band Nada Surf as he prepared a gourmet, sit-down dinner for 28 pals.

When I first invited Nada Surf to be on the show, I was told that Daniel Lorca was the guy I wanted to talk to. Th
response: "i'm way into it, but i don't want to talk about it, i wanna do it." After years of only having access to
their sound check and set, I've been doing a lot of *talking* about cooking with rockstars. To actually cook w
true.

Six-hour Salad

Daniel prepared a salad of arugula, smoked tomatoes, tomato jam, and grilled avocado (it's as good as it sounds!). I jokingly called it "6-hour Salad" because that's how long he worked on it. The fresh tomatoes were slowly smoked over woodchips in the grill, and when

The Box Model (cont'd)

- The **box model** is the foundation of CSS page layout.
- Apply properties such as **borders, margins, padding**, and **backgrounds** to element boxes.
- Position, move, grow, and shrink boxes to create fixed or flexible page layouts.

CSS Units of Measurement

CSS provides a variety ways to specify measurements:

Absolute units
Have predefined meanings or real-world equivalents

Relative units
Based on the size of something else, such as the default text size or the size of the parent element

Percentages
Calculated relative to another value, such as the size of the parent element

Absolute Units

With the exception of pixels, absolute units are not appropriate for web design:

`px`	pixel
`in`	inches
`mm`	millimeters
`cm`	centimeters
`q`	1/4 millimeter
`pt`	points (1/72 inch)
`pc`	pica (1 pica = 12 points = 1/6 inch)

Relative Units

Relative units are based on the size of something else:

`em`	a unit equal to the current font size
`ex`	x-height, equal to the height of a lowercase *x*
`rem`	root em, equal to the font size of the `html` element
`ch`	zero width, equal to the width of a zero (0)
`vw`	viewport width unit (equal to 1/100 of viewport width)
`vh`	viewport height unit (1/100 of viewport height)
`vmin`	viewport minimum unit (value of `vh` or `vw`, whichever is smaller)
`vmax`	viewport maximum unit (value of `vh` or `vw`, whichever is larger)

RELATIVE UNITS

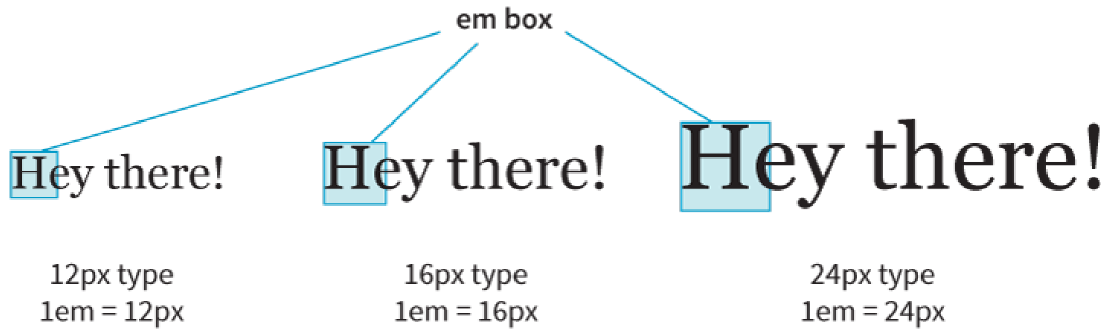
The rem Unit

- The **rem** (**root em**) unit is based on the font size of the **`html`** element, whatever that happens to be.
- **Default** in modern browsers: Root font size is 16 pixels, so a **rem = a 16-pixel unit.**
- If the root font size of the document changes, so does the size of a rem (and that’s good for keeping elements proportional).

RELATIVE UNITS

The em Unit

- The **em** unit is traditionally based on the width of a capital letter *M* in the font.
- When the font size is 16 pixels,1em = 16 pixels, 2em = 32 pixels, and so on.



NOTE: Because they're based on the font size of the current element, the size of an em may not be consistent across a page.

RELATIVE UNITS

Viewport Percentage Lengths (vw/vh)

Viewport width (**`vw`**) and viewport height (**`vh`**) units are relative to the size of the viewport (browser window):

`vh` = 1/100th width of viewport

`vh` = 1/100th height of viewport

They're useful for making an element fill the viewport or a specified percentage of it. This image will be 50% the width and height of the viewport:

```
img { width: 50vw; height: 50vh; }
```

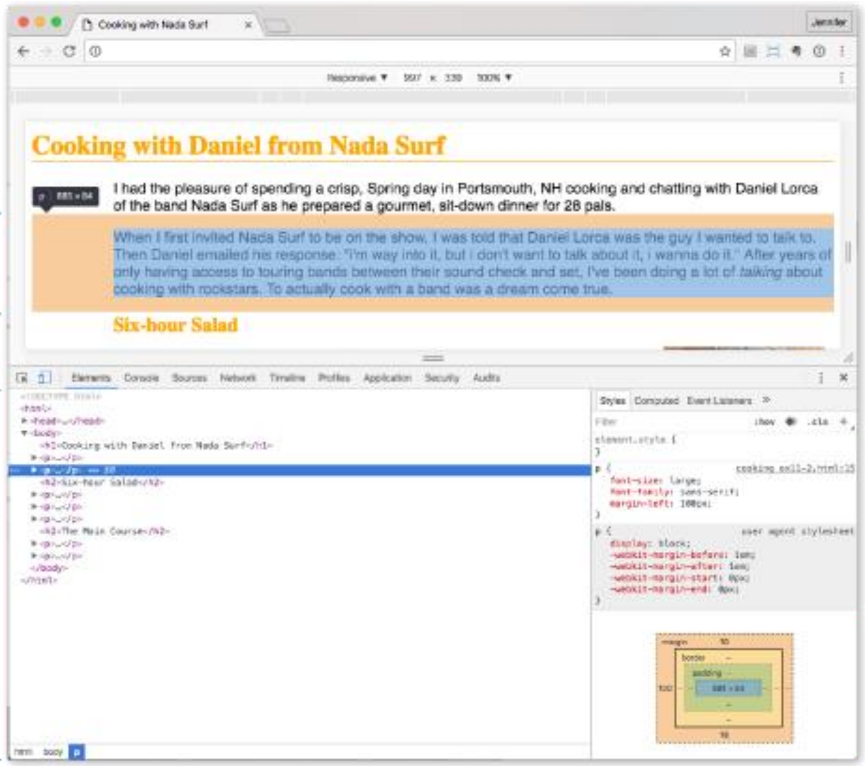
Browser Developer Tools

Major browsers have built-in tools that aid development:

- HTML, CSS, and JavaScript inspectors
- Network speed reports
- Animation tools
- Other helpful features

Browser Developer Tools (cont'd)

Chrome DevTools (View > Developer > Developer Tools)



Cooking with Daniel from Nada Surf

I had the pleasure of spending a crisp, Spring day in Portsmouth, NH cooking and chatting with Daniel Lorca of the band Nada Surf as he prepared a gourmet, sit-down dinner for 28 pals.

When I first invited Nada Surf to be on the show, I was told that Daniel Lorca was the guy I wanted to talk to. Then Daniel emailed his response: "I'm way into it, but i don't want to talk about it, i wanna do it." After years of only having access to touring bands between their sound check and set, I've been doing a lot of *talking* about cooking with rockstars. To actually cook with a band was a dream come true.

Six-hour Salad

```
<!DOCTYPE html>
<html>
▶ <head>…</head>
▼ <body>
    <h1>Cooking with Daniel from Nada Surf</h1>
  ▶ <p>…</p>
  ▶ <p>…</p> == $0
    <h2>Six-hour Salad</h2>
  ▶ <p>…</p>
  ▶ <p>…</p>
  ▶ <p>…</p>
    <h2>The Main Course</h2>
  ▶ <p>…</p>
  ▶ <p>…</p>
  </body>
</html>
```

```
element.style {
}
p {                                cooking_ex11-2.html:15
  font-size: large;
  font-family: sans-serif;
  margin-left: 100px;
}
p {                                user agent stylesheet
  display: block;
  -webkit-margin-before: 1em;
  -webkit-margin-after: 1em;
  -webkit-margin-start: 0px;
  -webkit-margin-end: 0px;
}
```

Elements selected in code are highlighted in the browser view.

HTML source for the page.

All styles that are applied to the selected element.

Margins, borders, and paddings applied to the element.

Firefox, Safari, Opera, and Microsoft Edge also have developer tools.

12
FORMATTING TEXT

- Font-related properties
- Text line settings
- Various text effects
- List style properties
- ID, class, and descendent selectors
- Specificity

Designing Text

Styling text on the web is tricky because you don’t have control over how the text displays for the user:

- They may not have the font you specify.
- They may have their text set larger or smaller than you designed it.

Best practices allow for flexibility in text specification.

Typesetting Terminology

- A typeface is a set of characters with a single design (example: Garamond).
- A font is a particular variation of the typeface with a specific weight, slant, or ornamentation (example: Garamond Bold Italic).
- In traditional metal type, each size was a separate font (example: 12-point Garamond Bold Italic).
- On a computer, fonts are generally stored in individual font files.

CSS Basic Font Properties

CSS font properties deal with specifying the shapes of the characters themselves:

- `font-family`
- `font-size`
- `font-weight`
- `font-style`
- `font-variant`
- `font` (a shorthand that includes settings for all of the above)

Specifying the Font Family

font-family

Values: One or more font family names, separated by commas

Example:

```
body { font-family: Arial; }
var    {    font-family:    Courier,
monospace; }
```

Specifying the Font Family (cont'd)

- Font names must be capitalized (except generic font families).
- Use commas to separate multiple font names.
- If the name has a character space, it must appear within quotation marks:

```
p { font-family: "Duru Sans", Verdana,
sans-serif; }
```

Using Fonts on the Web

- The font must be available on the user's machine for it to display.
- The best practice is to provide a list of options. The browser uses the first one that is available.
- Start with the font you want and then provide backup options ending with a generic font family, as shown here:

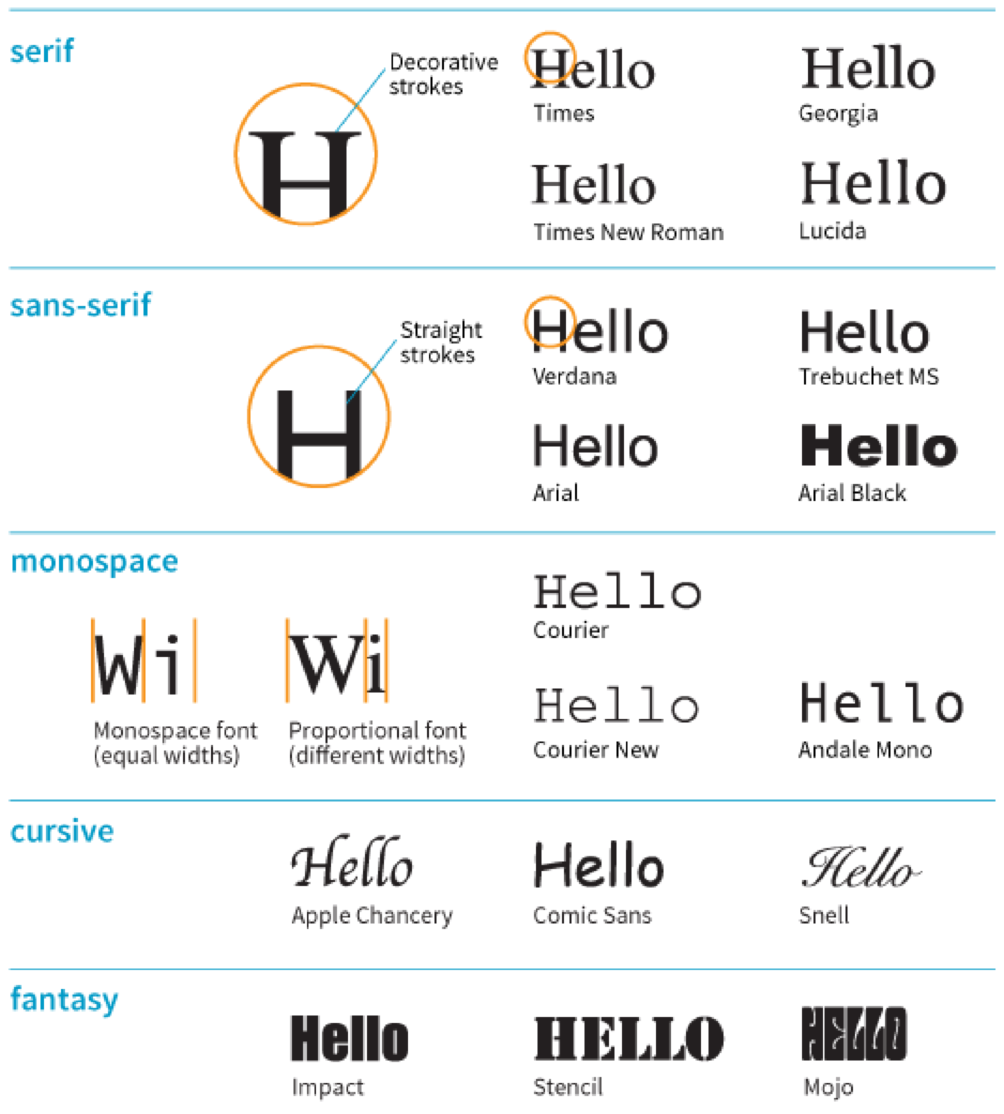
  ```
  p { font-family: "Duru Sans", Verdana, sans-serif; }
  ```

- You can also download a web font with the page, but it adds to the download and display time.

Generic Font Families

- Generic font families instruct the browser to use an available font from one of five stylistic categories:
  serif, sans-serif, monospace, cursive, fantasy
- Generic font families are often used as the last backup option.

Generic Font Families (cont'd)



Specifying Font Size

`font-size`

Values:

- CSS length units
- Percentage value
- Absolute keywords (`xx-small`, `x-small`, `small`, `medium`, `large`, `x-large`, `xx-large`)
- Relative keywords (`larger`, `smaller`)

Example:

```
h1 { font-size: 2.5rem; }
```

Specifying Font Size (cont'd)

Most common sizing methods:

- **rem** and **em** units
- **percentages** (based on the inherited font size for that element)
- **pixels** (px) can be used, but they're not flexible.

Font Size: rem Units

- The **rem (root em)** is equal to the font size of the **`html`** (root) element.
- In browsers, the **default** root size is 16 pixels, so: **1 rem = 16 pixels.**
- If the font size of the root is changed, rem measurements change too.
- !!! Old browsers *do not* support rem units (IE8 and earlier).

Font Size: em Units

- The em unit is based on the current font size of the element.
- The default font size is 16 pixels. By default, 1em = 16 pixels.
- But if you change the font size of the element, the size of its em unit changes too.
- Ems may be different sizes in different parts of the document and may compound larger or smaller when elements are nested.
- This makes ems a little tricky to use, although they are better supported than rem units.

Font Weight (Boldness)

`font-weight`

Values: `normal`, `bold`, `bolder`, `lighter`, `100`, `200`, `300`, `400`, `500`, `600`, `700`, `800`, `900`

Example:

```
h1 { font-weight: normal; }
span.new { font-weight: bold; }
```

- Most common values are normal and bold.
- Numerical values are useful when using a font with multiple weights.

Font Style (Italics)

`font-style`

Values: `normal`, `italic`, `oblique`

Example:

```
cite { font-style: italic; }
```

- Makes text italic, normal, or oblique (slanted, but generally the same as italics).

Small Caps

`font-variant`

Values (in CSS2.1): `normal`, `small-caps`

Example:

```
abbr { font-variant: small-caps; }
```

- Small caps are a separate font design that uses small uppercase characters in place of lowercase letters.
- They help acronyms and other strings of capital letters blend in with the weight of the surrounding text.

Condensed and Extended Text

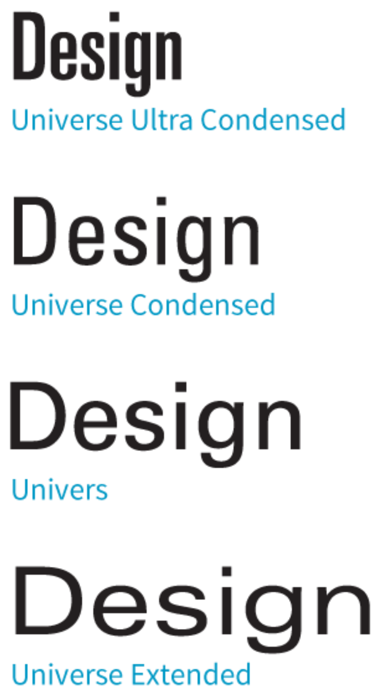
`font-stretch`

Values (in CSS2.1): `normal`, `ultra-condensed`, `extra-condensed`, `condensed`, `semi-condensed`, `semi-expanded`, `expanded`, `extra-expanded`, `ultra-expanded`

Example:

```
abbr { font-variant: small-caps; }
```

- Tells the browser to select a normal, condensed, or extended font variation from a typeface if it is available



The Shortcut font Property

`font`

Values (in CSS2.1): A list of values for all the individual properties, in this order:

```
{font: style weight stretch variant size/line-
height font-family}
```

At minimum, it must contain **`font-size`** and **`font-family`**, in that order. Other values are optional and may appear in any order prior to **`font-size`**.

Example:

```
 p { font: 1em sans-serif; }
 h3 { font: oblique bold small-caps 1.5em
Verdana, sans-serif; }
```

Advanced Typography

The CSS3 Font Module offers properties for fine-tuned typography control, including:

- Ligatures
- Superscript and subscript
- Alternate characters (such as a swash design for an *S*)
- Proportional font sizing using x-height
- Kerning
- OpenType font features

Text Line Treatments

Some properties control whole lines of text:

- Line height (`line-height`)
- Indents (`text-indent`)
- Horizontal alignment (`text-align`)

Line Height

`line-height`

Values: *Number*, *length*, *percentage*, `normal`

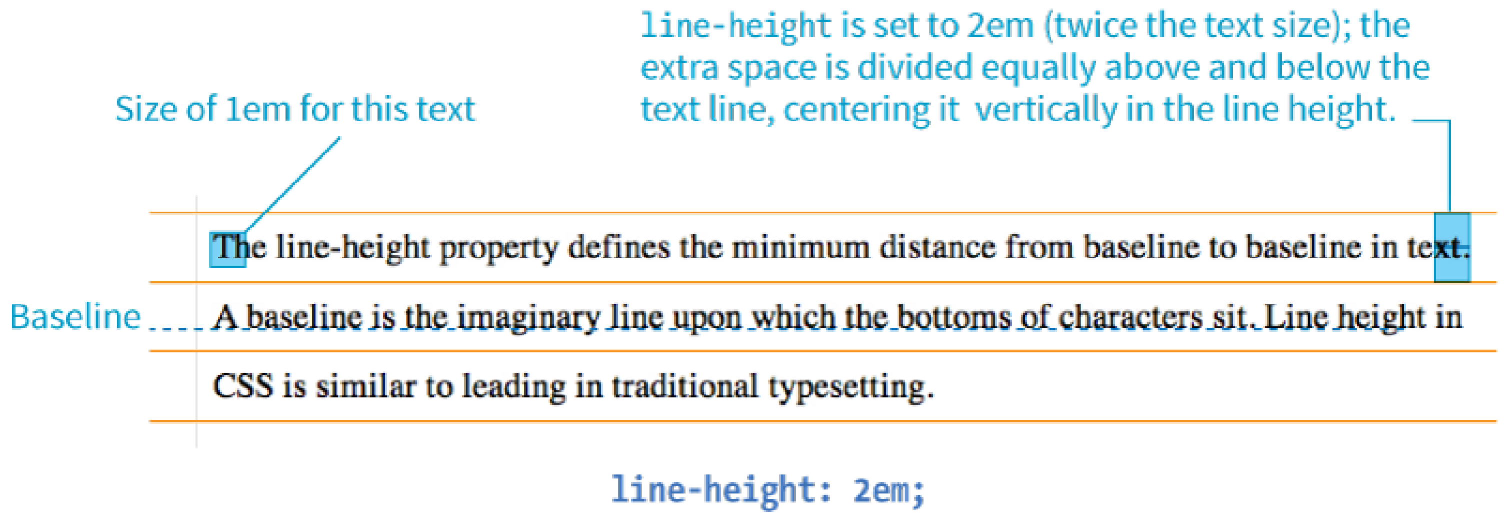
Example:

```
p { line-height: 1.4em; }
```

- Line height defines the minimum distance from baseline to baseline in text.

Line Height (cont'd.)

- The **baseline** is the imaginary line upon which the bottoms of characters sit.
- If a large character or image is on a line, the line height expands to accommodate it.


line-height is set to 2em (twice the text size); the extra space is divided equally above and below the text line, centering it vertically in the line height.
Size of 1em for this text
Baseline
line-height: 2em;

The line-height property defines the minimum distance from baseline to baseline in text. A baseline is the imaginary line upon which the bottoms of characters sit. Line height in CSS is similar to leading in traditional typesetting.

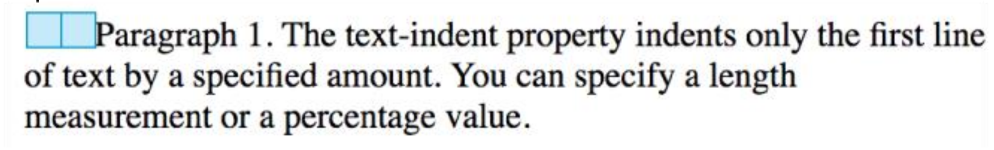
Indents

`text-indent`

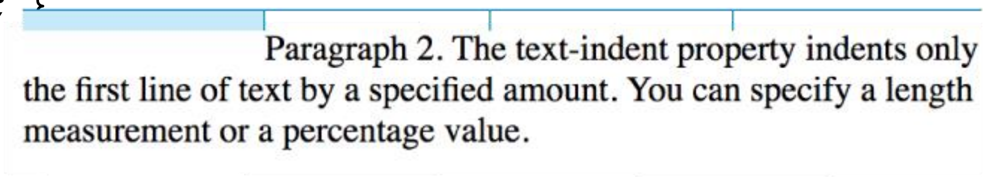
Values: *Length, percentage*

Examples:

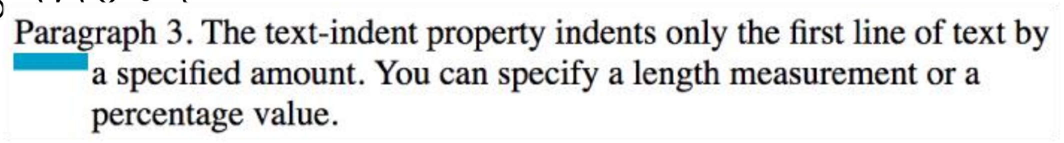
```
p {text-ident: 2em;}
```


Paragraph 1. The text-indent property indents only the first line of text by a specified amount. You can specify a length measurement or a percentage value.

```
p {text-ident: 25%;}
```


Paragraph 2. The text-indent property indents only the first line of text by a specified amount. You can specify a length measurement or a percentage value.

```
p {text-ident: -35px;}
```

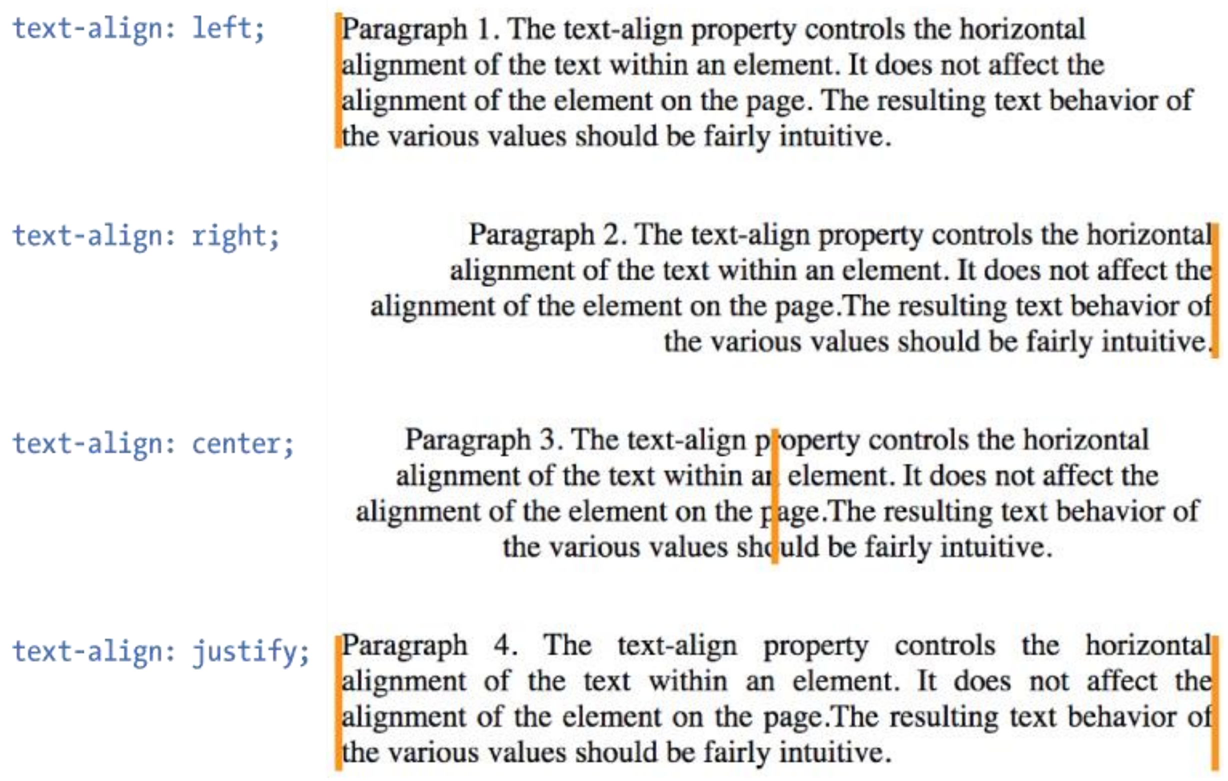

Paragraph 3. The text-indent property indents only the first line of text by a specified amount. You can specify a length measurement or a percentage value.

Horizontal Text Alignment

text-align

Values: `left`, `right`, `center`, `justify`, `start`, `end`

Examples:



text-align: left;

Paragraph 1. The text-align property controls the horizontal
alignment of the text within an element. It does not affect the
alignment of the element on the page. The resulting text behavior of
the various values should be fairly intuitive.

text-align: right;

Paragraph 2. The text-align property controls the horizontal
alignment of the text within an element. It does not affect the
alignment of the element on the page.The resulting text behavior of
the various values should be fairly intuitive.

text-align: center;

Paragraph 3. The text-align property controls the horizontal
alignment of the text within an element. It does not affect the
alignment of the element on the page.The resulting text behavior of
the various values should be fairly intuitive.

text-align: justify;

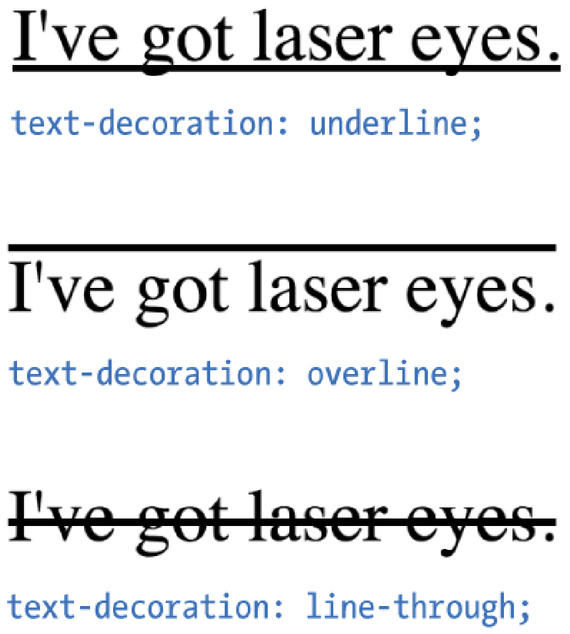
Paragraph 4. The text-align property controls the horizontal
alignment of the text within an element. It does not affect the
alignment of the element on the page.The resulting text behavior of
the various values should be fairly intuitive.

Underlines (Text Decoration)

`text-decoration`

Values: `none`, `underline`, `overline`, `line-through`, `blink`

Examples:



I've got laser eyes.

`text-decoration: underline;`

I've got laser eyes.

`text-decoration: overline;`

~~I've got laser eyes.~~

`text-decoration: line-through;`

NOTE: `text-decoration` is often used to turn off underlines under links:

```
a {
  text-decoration: none;
}
```

Text Decoration Tips

- If you turn off underlines under links, be sure there is another visual cue to compensate.
- Underlining text that is not a link may be misleading. Consider italics instead.
- Don’t use `blink`. Browsers don’t support it anyway.

- Got here fall 2019

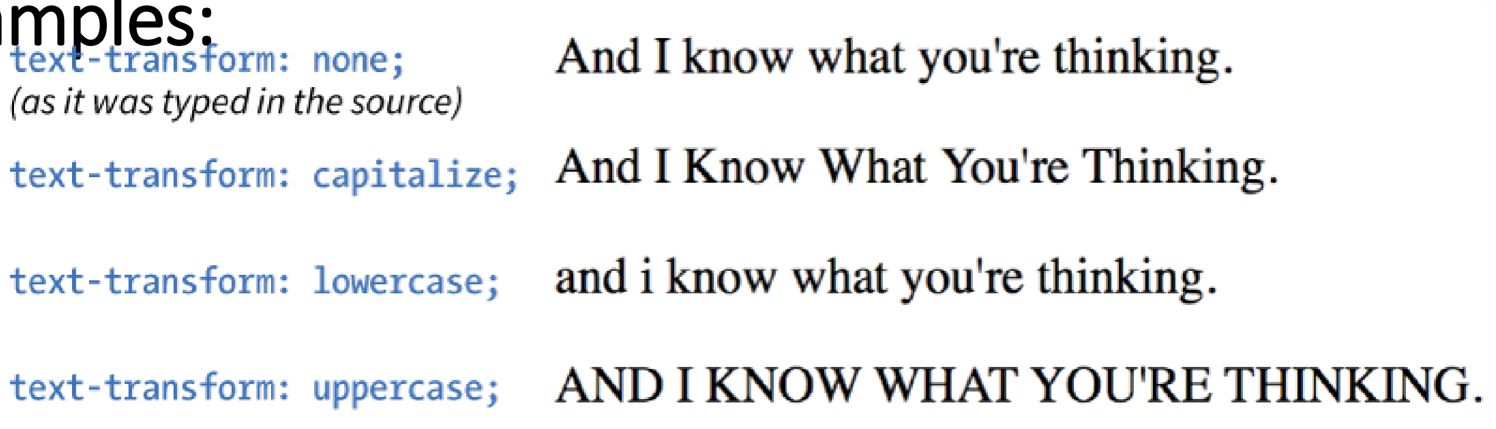
Capitalization

`text-transform`

Values:

`none`, `capitalize`, `lowercase`, `uppercase`, `full-width`

Examples:



`text-transform: none;` *(as it was typed in the source)*	And I know what you're thinking.
`text-transform: capitalize;`	And I Know What You're Thinking.
`text-transform: lowercase;`	and i know what you're thinking.
`text-transform: uppercase;`	AND I KNOW WHAT YOU'RE THINKING.

Spacing

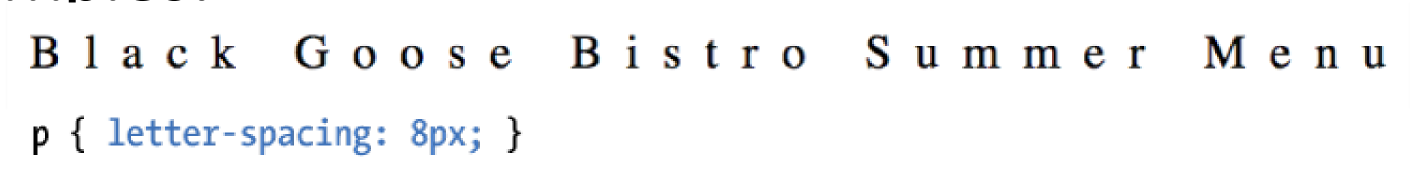
`letter-spacing`

Values: *length*, `normal`

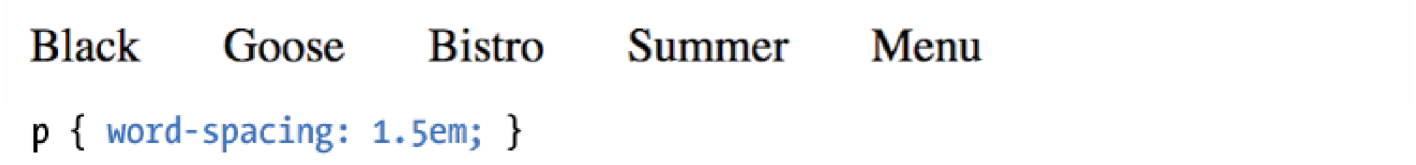
`word-spacing`

Values: *length*, `normal`

Examples:


B l a c k   G o o s e   B i s t r o   S u m m e r   M e n u

```
p { letter-spacing: 8px; }
```


Black   Goose   Bistro   Summer   Menu

```
p { word-spacing: 1.5em; }
```

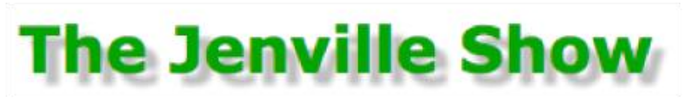
Text Shadow

`text-shadow`

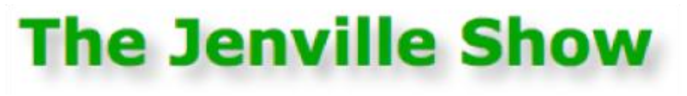
Values: *'horizontal-offset' 'vertical-offset' 'blur-radius' 'color'*, `none`

The value is two offset measurements, an optional blur radius, and a color value (with no commas between).

Example:



```
text-shadow: .2em .2em .1em silver;
```



```
text-shadow: .2em .2em .3em silver;
```

List Style Properties

There are three properties for affecting the display of lists:

- **`list-style-type`**
  Chooses the type of list marker
- **`list-style-position`**
  Sets the position of the marker relative to the list element box
- **`list-style-image`**
  Allows you to specify your own image for use as a bullet

LIST STYLES

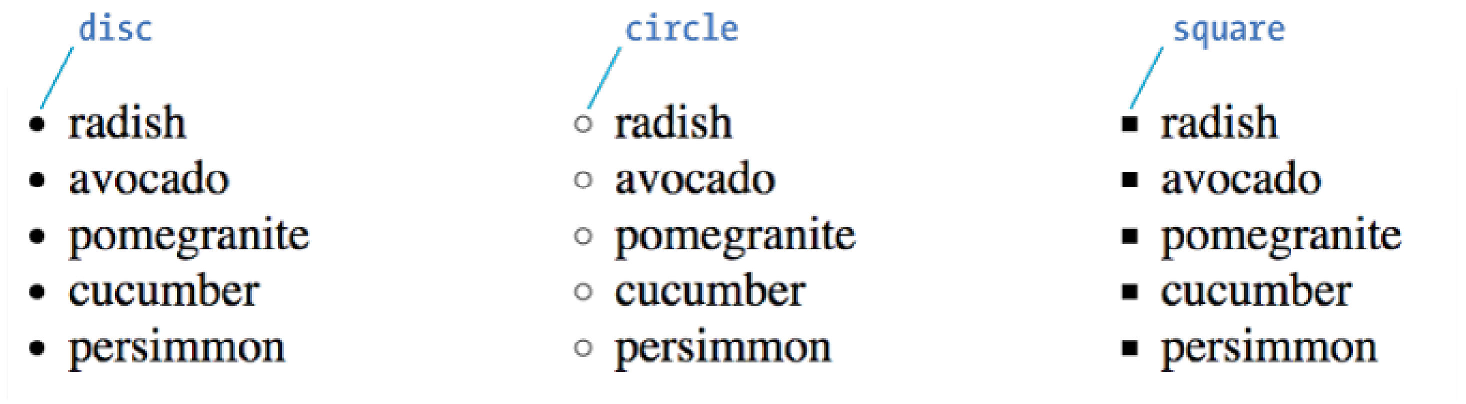
Choosing a Marker

`list-style-type`

Values:

`none`, `disc`, `circle`, `square`, `decimal`, `decimal-leading-zero`, `lower-alpha`, `upper-alpha`, `lower-latin`, `upper-latin`, `lower-roman`, `upper-roman`, `lower-greek`

Unordered lists: `ul { list-style-type: keyword; }`



LIST STYLES

Choosing a Marker (cont’d)

Ordered lists: `ol { list-style-type: keyword; }`

Keyword	System
`decimal`	1, 2, 3, 4, 5…
`decimal-leading-zero`	01, 02, 03, 04, 05…
`lower-alpha`	a, b, c, d, e…
`upper-alpha`	A, B, C, D, E…
`lower-latin`	a, b, c, d, e… (same as lower-alpha)
`upper-latin`	A, B, C, D, E… (same as upper-alpha)
`lower-roman`	i, ii, iii, iv, v…
`upper-roman`	I, II, III, IV, V…

LIST STYLES

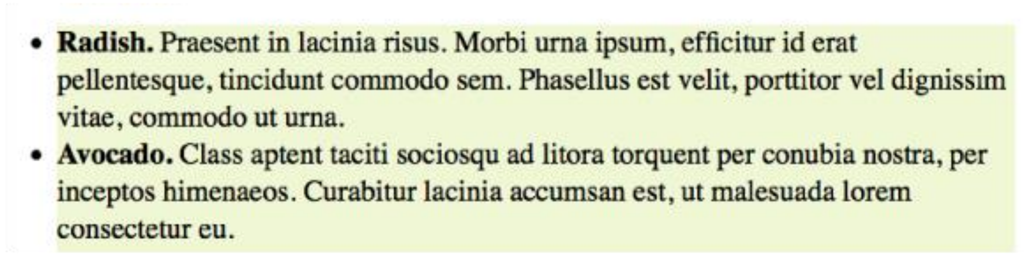
Marker Position

`list-style-position`

Values: `inside`, `outside`, `hanging`

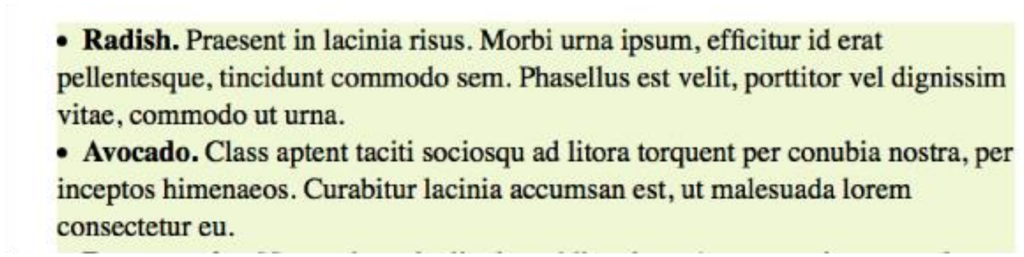
Positions the marker relative to the content area:

`outside`



- **Radish.** Praesent in lacinia risus. Morbi urna ipsum, efficitur id erat pellentesque, tincidunt commodo sem. Phasellus est velit, porttitor vel dignissim vitae, commodo ut urna.
- **Avocado.** Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Curabitur lacinia accumsan est, ut malesuada lorem consectetur eu.

`inside`



• **Radish.** Praesent in lacinia risus. Morbi urna ipsum, efficitur id erat pellentesque, tincidunt commodo sem. Phasellus est velit, porttitor vel dignissim vitae, commodo ut urna.
• **Avocado.** Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Curabitur lacinia accumsan est, ut malesuada lorem consectetur eu.

LIST STYLES

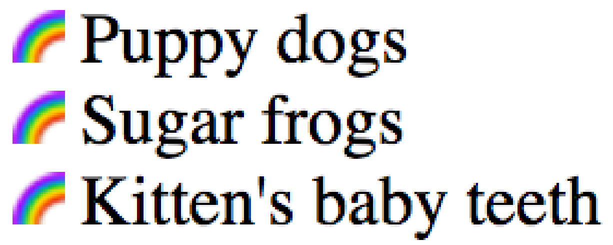
Custom Bullets

list-style-image

Values: `url(`*location*`), none`

Example:

```
  ul {
      list-style-type: disc;
      list-style-image:
  url(/images/rainbow.gif);
      list-st                           ide;
    }
```



Puppy dogs
Sugar frogs
Kitten's baby teeth

More Selector Types

- Descendent selectors
- ID selectors
- Class selectors
- Universal selector

Descendent Selectors

A **descendent selector** targets elements contained in another element.

It's a kind of **contextual selector** (it selects based on relationship to another element).

It's indicated in a list separated by a character space.

```
ol a {font-weight: bold;}
```

(only the links (**`a`**) in ordered lists (**`ol`**) would be bold)

```
h1 em {color: red;}
```

(only emphasized text in h1s would be red)

Descendent Selectors (cont'd)

They can appear as part of a grouped selector:

```
h1 em, h2 em, h3 em {color: red;}
```

(only emphasized text in `h1`, `h2`, and `h3` elements)

They can be several layers deep:

```
ol a em {font-variant: small-caps;}
```

(only emphasized text in links in ordered lists)

ID Selectors

ID selectors (indicated by a # symbol) target elements based on the value of their ID attributes:

```
<li id="primary">Primary color t-shirt</li>
```

To target just that item:

```
li#primary {color: olive;}
```

To omit the element name:

```
#primary {color: olive;}
```

It can be used as part of a compound or contextual selector:

```
#intro a { text-decoration: none;}
```

Class Selectors

Class selectors (indicated by a . symbol) select elements based on the value of their class attributes:

```
p.special { color: orange;}
```

(All paragraphs with the class name "special" would be orange.)

To target *all* element types that share a class name, omit the element name in the selector:

```
.hilight { background-color: yellow;}
```

(All elements with the class “hilight” would have a yellow background.)

Universal Selector

The universal element selector (*****) matches any element, like a wildcard in programming languages:

```
* {border: 1px solid gray;}
```

(puts a 1-pixel gray border around every element in the document)

Can be used as part of contextual selectors:

```
#intro * {border: 1px solid gray;}
```

(selects all elements contained within an element with the ID **`intro`**)

Specificity Basics

Specificity refers to a system for sorting out which selectors have more weight when resolving style rule conflicts.

More specific selectors have more weight.

In simplified terms, it works like this:

- Inline styles with the `style` attribute are more specific than (and will override…)
- ID selectors, which are more specific than (and will override…)
- Class selectors, which are more specific than (and will override…)
- Individual element selectors

Calculating Specificity

There is a system used to calculate specificity. Start by drawing three boxes:

[ ] [ ] [ ]

For each style rule:

1. Count the IDs in the selector and put that number in the first box.
2. Count the class and pseudo-class selectors and put the number in the second box.
3. Count the element names and put the number in the third box

[ID] [class] [elements]

4. The first box that is not a tie determines which selector wins.

Calculating Specificity (cont'd)

Example:

```
h1 { color: red;}                 [0]  [0]  [1]

h1.special { color: lime; }   [0]  [1]  [1]
```

The second one has a class selector and the first one doesn't, therefore the second one is more specific and has more weight.

The lime color applies to `h1`s when they have the class name "special."

Using Specificity

Use specificity strategically to take advantage of overrides:

```
p { line-height: 1.2em; }                [0]   [0]
[1]
```

(sets the line-height for all paragraphs)

```
blockquote p { line-height: 1em; }[0]    [0]
[2]
```

(more specific selector changes line-height when the paragraph is in a blockquote)

```
p.intro { line-height: 2em; }            [0]   [1]
[1]
```

(paragraphs with the class "intro" have a line-height of 2em, even when they're in a blockquote. A class name in the selector has more weight than two element names.)

13
COLORS AND BACKGROUNDS

- CSS color names
- RGB and HSL color values
- Foreground and background colors
- Tiling background images
- More selectors and external style sheets

Named Color Values

Specify foreground or background color using one of 140 predefined CSS3 color names:

```
h1 { color: red; }
h2 { color: darkviolet; }
body { background-color:
papayawhip; }
```

learningwebdesign.com/colornames.html

Numeric Color Values

For more control, define colors numerically using one of these color models:

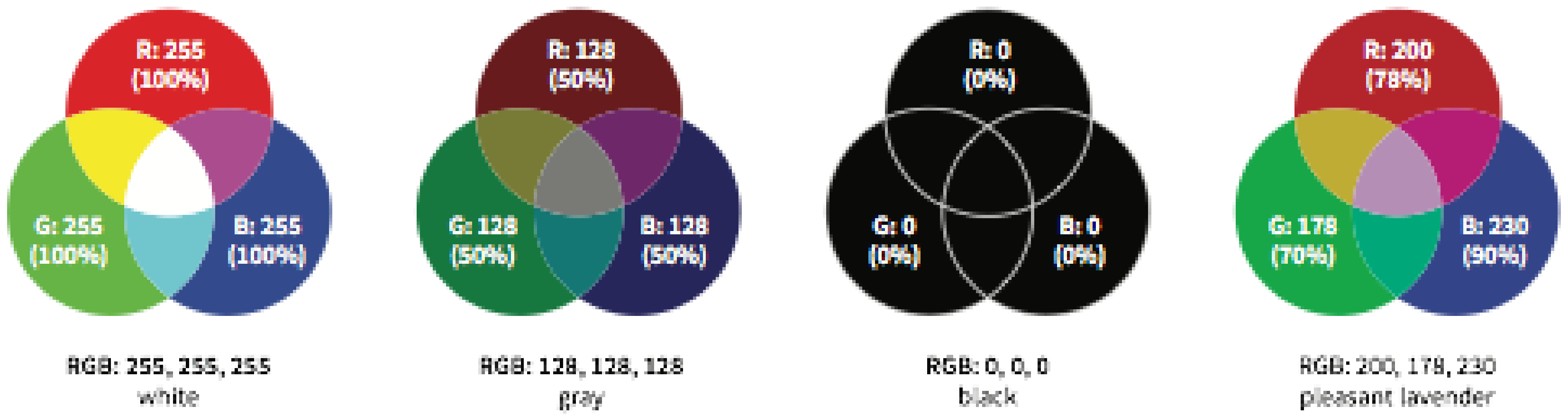
- RGB (combination of red, green, and blue light values)
- RGBa (RGB plus alpha transparency)
- HSL (hue, saturation, and luminosity)
- HSLa (HSL plus alpha transparency)

RGB Color

The **RGB color model** mixes color with red, green, and blue light.

Each channel can have 256 shades, for millions of color options.

The RGB Color Model



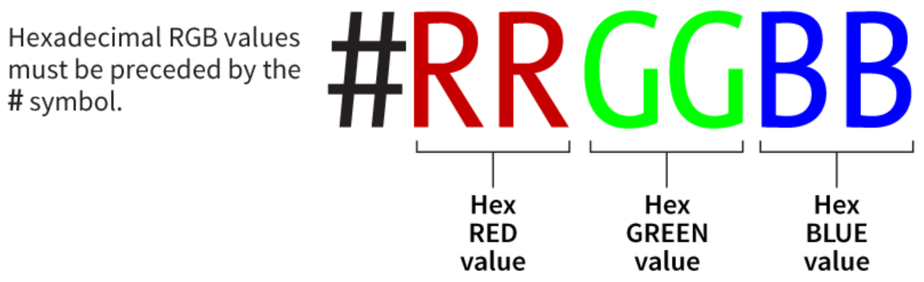
RGB Values in Style Rules

There are four formats for providing RGB color values:

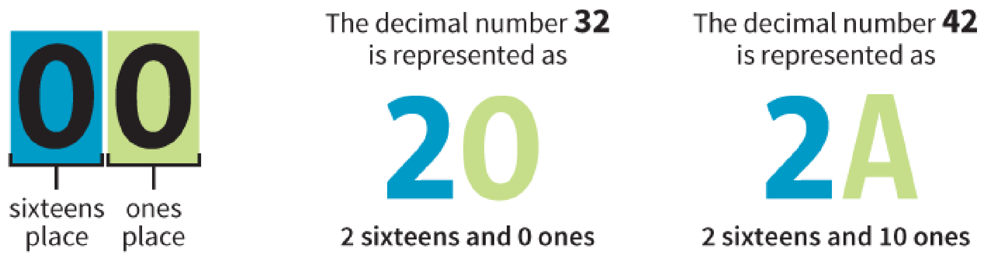
- RGB values (0 to 255): `rgb(200,178,230)`
- Percentage values: `rgb(78%,70%,90%)`
- Hexadecimal values: `#C8B2E6`
- Condensed hexadecimal values (for double-digits only): `#F06` is the same as `#FF0066`

Hexadecimal RGB Values

Red, green, and blue values converted to hexadecimal and preceded by the # symbol.

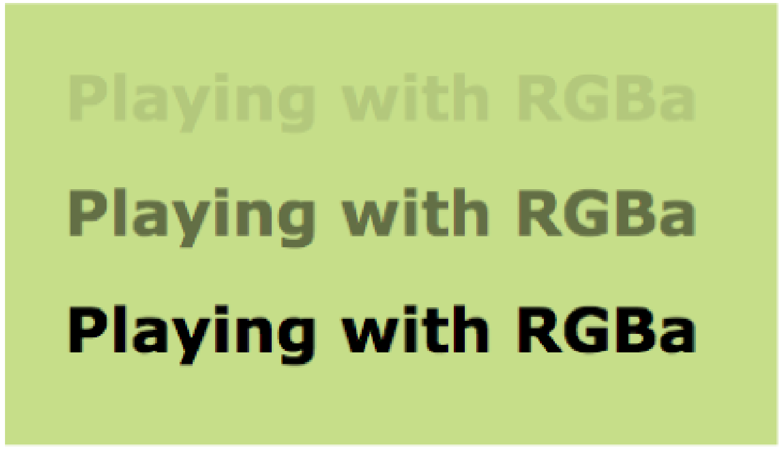


Decimal	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F


00
sixteens place
ones place
The decimal number 32 is represented as
20
2 sixteens and 0 ones
The decimal number 42 is represented as
2A
2 sixteens and 10 ones

RGBa Color

- RGB + an alpha channel for transparency
- The first three values are RGB. The fourth is the transparency level from 0 (transparent) to 1 (opaque).



```
color: rgba(0, 0, 0, .1);

color: rgba(0, 0, 0, .5);

color: rgba(0, 0, 0, 1);
```

HSL and HSLa

- Colors described by values for **hue** (°), **saturation** (%), and **luminosity** (%):

```
hsl(180,50%,75%)
```

- Hue specifies the position on a color wheel (in degrees) that has red at 0°, green at 120°, and blue at 240°.
- HSL is less commonly used than RGB, but some find it more intuitive.
- HSLa adds an alpha value for transparency.

Foreground Color

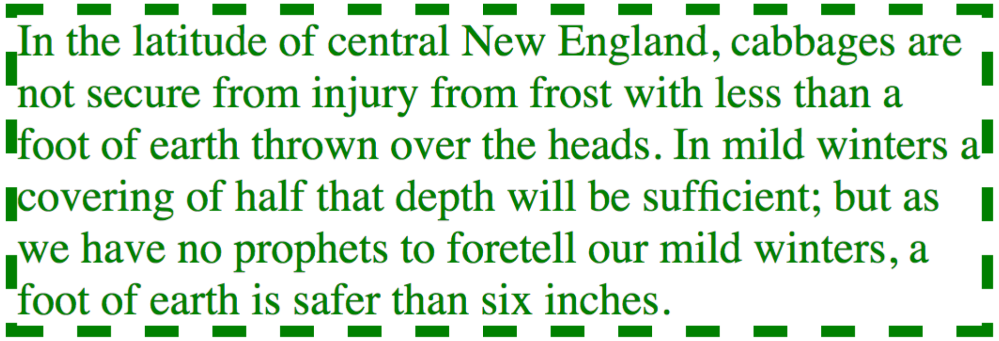
`color`

Values: *Color value* (named or numeric)

Example: `blockquote {border: 4px dashed; color: green;}`

The foreground of an element consists of its text and border

(if one is :



In the latitude of central New England, cabbages are not secure from injury from frost with less than a foot of earth thrown over the heads. In mild winters a covering of half that depth will be sufficient; but as we have no prophets to foretell our mild winters, a foot of earth is safer than six inches.

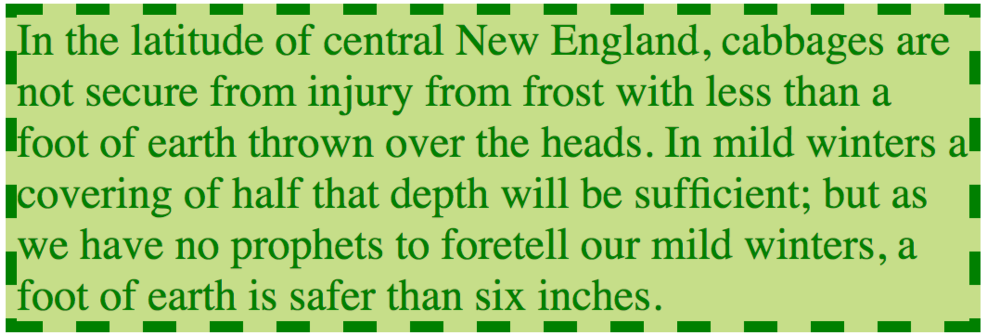
Background Color

background-color

Values: *Color value* (named or numeric)

Example:
```
blockquote {border: 4px dashed;
            background-color: green;}
```

The background painting area of an element fills the area behind the text to the outer edge of the border.

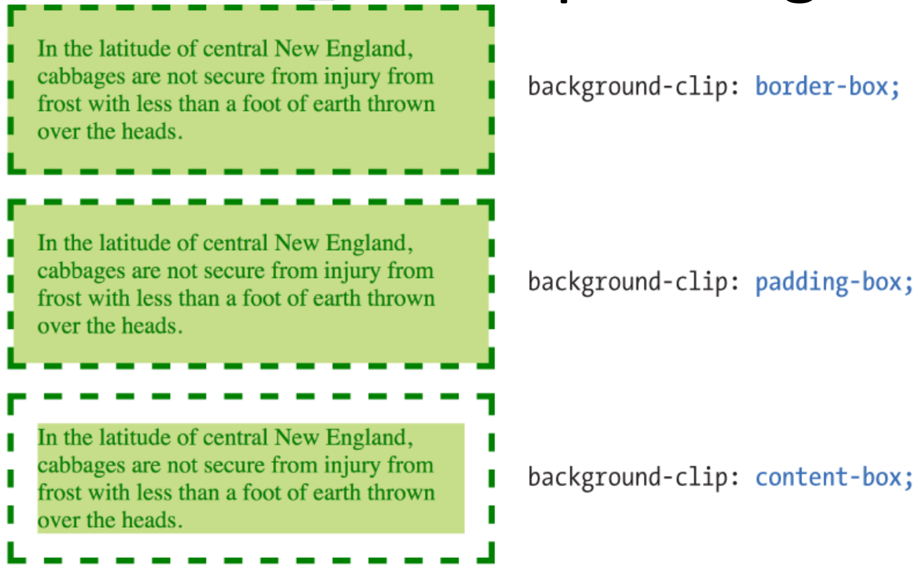

In the latitude of central New England, cabbages are not secure from injury from frost with less than a foot of earth thrown over the heads. In mild winters a covering of half that depth will be sufficient; but as we have no prophets to foretell our mild winters, a foot of earth is safer than six inches.

Clipping the Background

`background-clip`

Values: `border-box`, `padding-box`, `content-box`

Specifies where the background painting area ends.


In the latitude of central New England, cabbages are not secure from injury from frost with less than a foot of earth thrown over the heads.

```
background-clip: border-box;
```

In the latitude of central New England, cabbages are not secure from injury from frost with less than a foot of earth thrown over the heads.

```
background-clip: padding-box;
```

In the latitude of central New England, cabbages are not secure from injury from frost with less than a foot of earth thrown over the heads.

```
background-clip: content-box;
```

Opacity

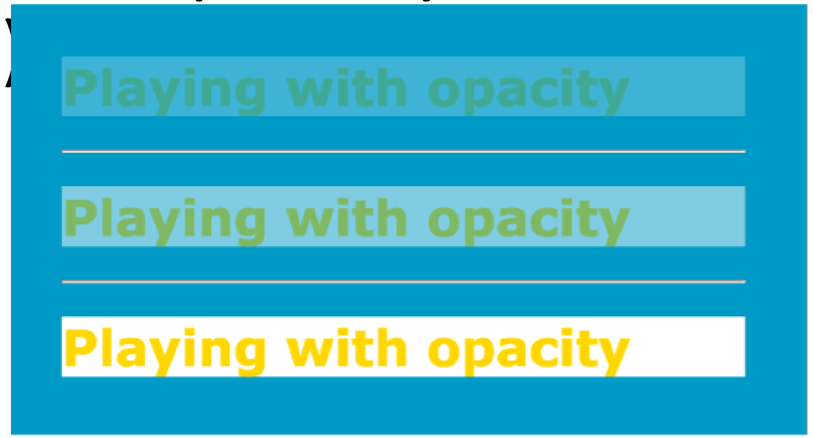
`opacity`

Values: *number* (0 to 1)

Example:

```
h1 {color: gold; background: white; opacity: .25;}
```

Specifies the transparency level from 0 (transparent)



```
opacity: .25;

opacity: .5;

opacity: 1;
```

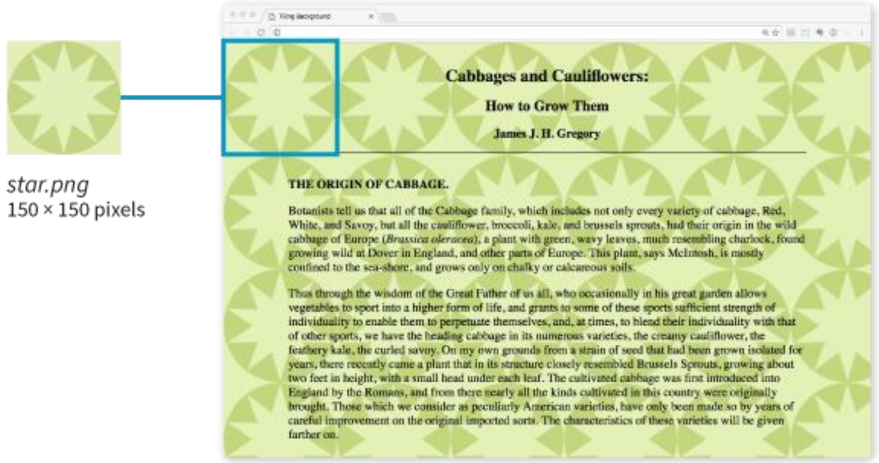
Tiling Background Images

`background-image`

Values: `url` (*location of image*), `none`

Example: `body {background-image: url(star.png);}`

Locates an image to be used as a tiling background image behind an element. By default, it starts at the top, left corner and repeats horizontally and vertically:



Cabbages and Cauliflowers:

How to Grow Them

James J. H. Gregory

THE ORIGIN OF CABBAGE.

Botanists tell us that all of the Cabbage family, which includes not only every variety of cabbage, Red, White, and Savoy, but all the cauliflower, broccoli, kale, and brussels sprouts, had their origin in the wild cabbage of Europe (*Brassica oleracea*), a plant with green, wavy leaves, much resembling charlock, found growing wild at Dover in England, and other parts of Europe. This plant, says McIntosh, is mostly confined to the sea-shore, and grows only on chalky or calcareous soils.

Thus through the wisdom of the Great Father of us all, who occasionally in his great garden allows vegetables to sport into a higher form of life, and grants to some of these sports sufficient strength of individuality to enable them to perpetuate themselves, and, at times, to blend their individuality with that of other sports, we have the heading cabbage in its numerous varieties, the creamy cauliflower, the feathery kale, the curled savoy. On my own grounds from a strain of seed that had been grown isolated for years, there recently came a plant that in its structure closely resembled Brussels Sprouts, growing about two feet in height, with a small head under each leaf. The cultivated cabbage was first introduced into England by the Romans, and from there nearly all the kinds cultivated in this country were originally brought. Those which we consider as peculiarly American varieties, have only been made so by years of careful improvement on the original imported sorts. The characteristics of these varieties will be given farther on.

Background Repeating

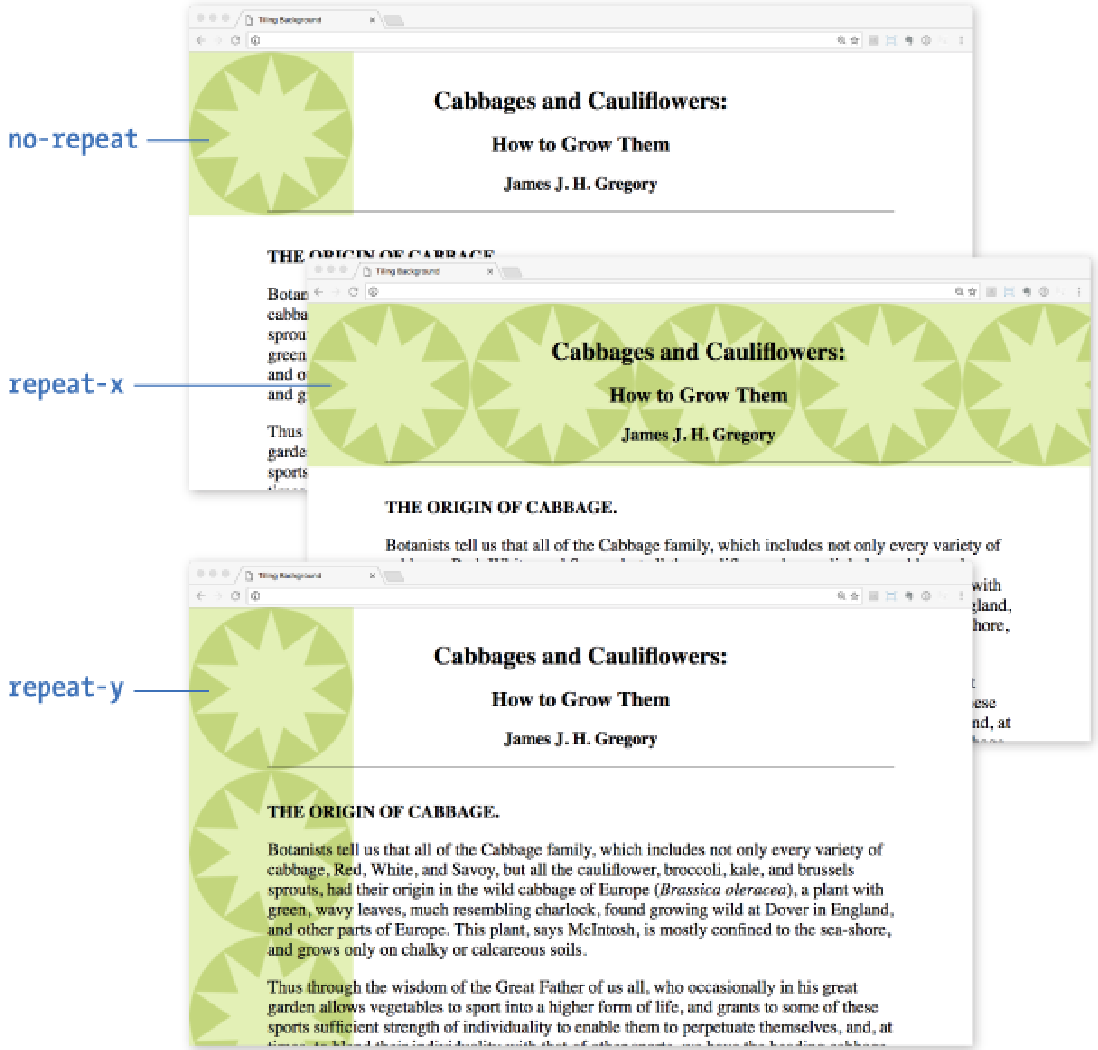
`background-repeat`

Values:
`repeat`, `no-repeat`, `repeat-x`, `repeat-y`, `space`, `round`

Specifies how the background image repeats and can restrict it to tiling in one direction or not at all:

- **`repeat-x`**: Tiles horizontally only
- **`repeat-y`**: Tiles vertically only
- **`space`**: Adds space around images so they fit in the window with no clipping
- **`round`**: Distorts the image so it fits without clipping

Background Repeating (cont'd)



Cabbages and Cauliflowers:

How to Grow Them

James J. H. Gregory

THE ORIGIN OF CABBAGE.

Botanists tell us that all of the Cabbage family, which includes not only every variety of cabbage, Red, White, and Savoy, but all the cauliflower, broccoli, kale, and brussels sprouts, had their origin in the wild cabbage of Europe (*Brassica oleracea*), a plant with green, wavy leaves, much resembling charlock, found growing wild at Dover in England, and other parts of Europe. This plant, says McIntosh, is mostly confined to the sea-shore, and grows only on chalky or calcareous soils.

Thus through the wisdom of the Great Father of us all, who occasionally in his great garden allows vegetables to sport into a higher form of life, and grants to some of these sports sufficient strength of individuality to enable them to perpetuate themselves, and, at

Background Position

`background-position`

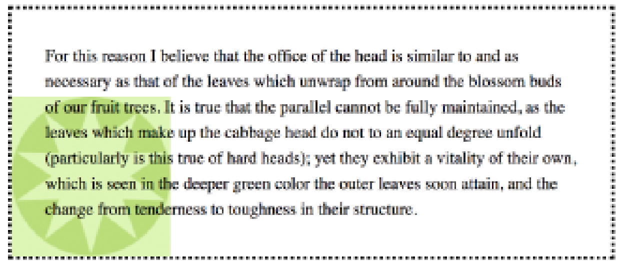
Values:
Length, *percentage*, `left`, `center`, `right`, `top`, `bottom`

Specifies the position of the origin image, the first image that is placed in the background from which tiling images extend.

Examples (horizontal position goes first):

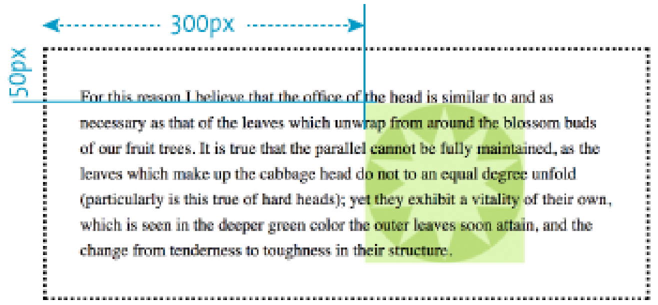
```
background-position: left bottom;
background-position: 300px 100px;
background-position: 25% 100%;
```

Background Position (cont'd)



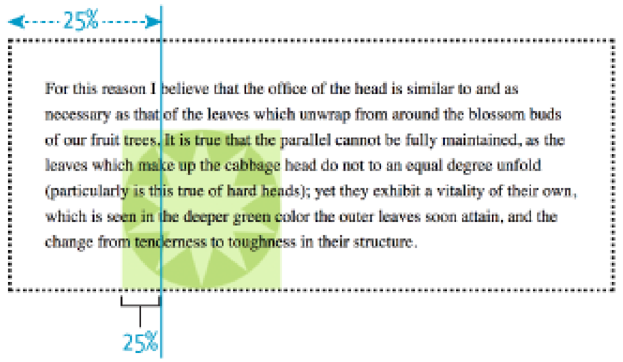
For this reason I believe that the office of the head is similar to and as necessary as that of the leaves which unwrap from around the blossom buds of our fruit trees. It is true that the parallel cannot be fully maintained, as the leaves which make up the cabbage head do not to an equal degree unfold (particularly is this true of hard heads); yet they exhibit a vitality of their own, which is seen in the deeper green color the outer leaves soon attain, and the change from tenderness to toughness in their structure.

```
background-position: left bottom;
```



For this reason I believe that the office of the head is similar to and as necessary as that of the leaves which unwrap from around the blossom buds of our fruit trees. It is true that the parallel cannot be fully maintained, as the leaves which make up the cabbage head do not to an equal degree unfold (particularly is this true of hard heads); yet they exhibit a vitality of their own, which is seen in the deeper green color the outer leaves soon attain, and the change from tenderness to toughness in their structure.

```
background-position: 300px 50px;
```



For this reason I believe that the office of the head is similar to and as necessary as that of the leaves which unwrap from around the blossom buds of our fruit trees. It is true that the parallel cannot be fully maintained, as the leaves which make up the cabbage head do not to an equal degree unfold (particularly is this true of hard heads); yet they exhibit a vitality of their own, which is seen in the deeper green color the outer leaves soon attain, and the change from tenderness to toughness in their structure.

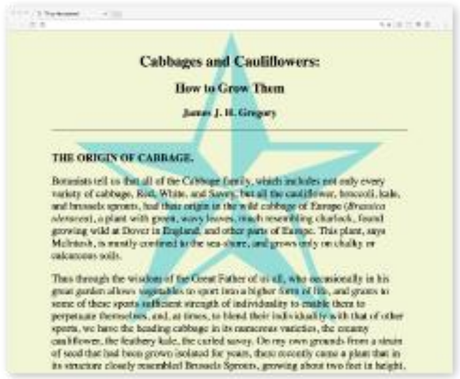
```
background-position: 25% 100%;
```

Background Attachment

`background-attachment`

Values: `scroll`, `fixed`, `local`

Specifies whether the background image scrolls with the content or stays in a fixed position relative to the viewport.

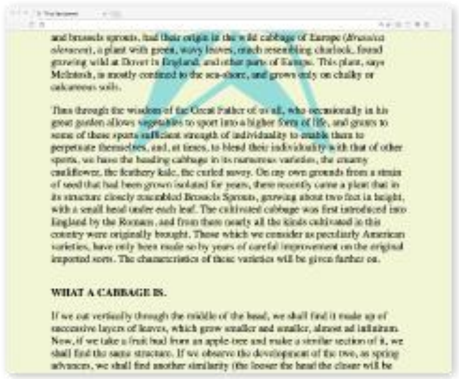

Cabbages and Cauliflowers:

How to Grow Them

James J. H. Gregory

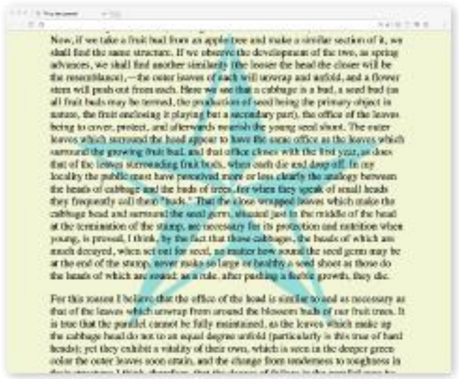
THE ORIGIN OF CABBAGE.

A large non-repeating background image in the **body** of the document.


WHAT A CABBAGE IS.

`background-attachment: scroll;`

By default, the background image is attached to the **body** element and scrolls off the page when the page content scrolls.


`background-attachment: fixed;`

When **`background-attachment`** is set to **`fixed`**, the image stays in its position relative to the browser viewing area and does not scroll with the content.

Background Size

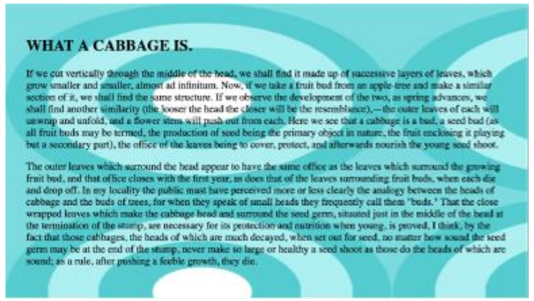
`background-size`

Values:

Length, *percentage*, `auto`, `cover`, `contain`

Specifies the size of the tiling image:

target.png
300 × 300 pixels

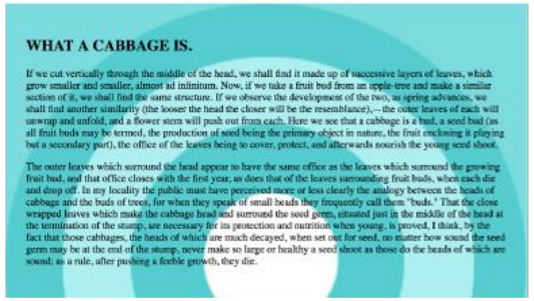


WHAT A CABBAGE IS.

If we cut vertically through the middle of the head, we shall find it made up of successive layers of leaves, which grow smaller and smaller, almost ad infinitum. Now, if we take a fruit bud from an apple-tree and make a similar section of it, we shall find the same structure. If we observe the development of the two, as spring advances, we shall find another similarity (the looser the head the closer will be the resemblance),—the outer leaves of each will unwrap and unfold, and a flower stem will push out from each. Here we see that a cabbage is a bud, a seed bud (as all fruit buds may be termed, the production of seed being the primary object in nature, the fruit enclosing it playing but a secondary part), the office of the leaves being to cover, protect, and afterwards nourish the young seed shoot.

The outer leaves which surround the head appear to have the same office as the leaves which surround the growing fruit bud, and that office closes with the first year, as does that of the leaves surrounding fruit buds, when each die and drop off. In my locality the public must have perceived more or less clearly the analogy between the heads of cabbage and the buds of trees, for when they speak of small heads they frequently call them "buds." That the close wrapped leaves which make the cabbage head and surround the seed germ, situated just in the middle of the head at the termination of the stump, are necessary for its protection and nutrition when young, is proved, I think, by the fact that those cabbages, the heads of which are much decayed, when set out for seed, no matter how sound the seed germ may be at the end of the stump, never make so large or healthy a seed shoot as those do the heads of which are sound; as a rule, after pushing a feeble growth, they die.

`background-size: 600px 300px;`



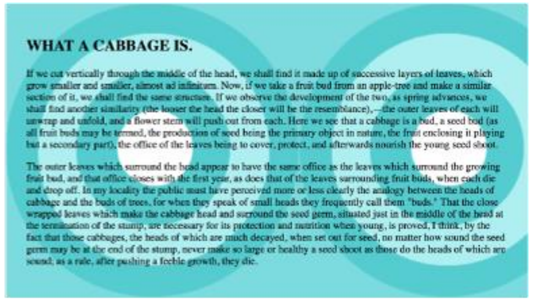
WHAT A CABBAGE IS.

If we cut vertically through the middle of the head, we shall find it made up of successive layers of leaves, which grow smaller and smaller, almost ad infinitum. Now, if we take a fruit bud from an apple-tree and make a similar section of it, we shall find the same structure. If we observe the development of the two, as spring advances, we shall find another similarity (the looser the head the closer will be the resemblance),—the outer leaves of each will unwrap and unfold, and a flower stem will push out from each. Here we see that a cabbage is a bud, a seed bud (as all fruit buds may be termed, the production of seed being the primary object in nature, the fruit enclosing it playing but a secondary part), the office of the leaves being to cover, protect, and afterwards nourish the young seed shoot.

The outer leaves which surround the head appear to have the same office as the leaves which surround the growing fruit bud, and that office closes with the first year, as does that of the leaves surrounding fruit buds, when each die and drop off. In my locality the public must have perceived more or less clearly the analogy between the heads of cabbage and the buds of trees, for when they speak of small heads they frequently call them "buds." That the close wrapped leaves which make the cabbage head and surround the seed germ, situated just in the middle of the head at the termination of the stump, are necessary for its protection and nutrition when young, is proved, I think, by the fact that those cabbages, the heads of which are much decayed, when set out for seed, no matter how sound the seed germ may be at the end of the stump, never make so large or healthy a seed shoot as those do the heads of which are sound; as a rule, after pushing a feeble growth, they die.

`background-size: cover;`

The entire background area of the element is covered, and the image maintains its proportions even if it is clipped.



WHAT A CABBAGE IS.

If we cut vertically through the middle of the head, we shall find it made up of successive layers of leaves, which grow smaller and smaller, almost ad infinitum. Now, if we take a fruit bud from an apple-tree and make a similar section of it, we shall find the same structure. If we observe the development of the two, as spring advances, we shall find another similarity (the looser the head the closer will be the resemblance),—the outer leaves of each will unwrap and unfold, and a flower stem will push out from each. Here we see that a cabbage is a bud, a seed bud (as all fruit buds may be termed, the production of seed being the primary object in nature, the fruit enclosing it playing but a secondary part), the office of the leaves being to cover, protect, and afterwards nourish the young seed shoot.

The outer leaves which surround the head appear to have the same office as the leaves which surround the growing fruit bud, and that office closes with the first year, as does that of the leaves surrounding fruit buds, when each die and drop off. In my locality the public must have perceived more or less clearly the analogy between the heads of cabbage and the buds of trees, for when they speak of small heads they frequently call them "buds." That the close wrapped leaves which make the cabbage head and surround the seed germ, situated just in the middle of the head at the termination of the stump, are necessary for its protection and nutrition when young, is proved, I think, by the fact that those cabbages, the heads of which are much decayed, when set out for seed, no matter how sound the seed germ may be at the end of the stump, never make so large or healthy a seed shoot as those do the heads of which are sound; as a rule, after pushing a feeble growth, they die.

`background-size: contain;`

The image is sized proportionally so it fits entirely in the element. There may be room left over for tiling (as shown).

Shorthand background Property

`background`

Values:
background-color background-image background-repeat background-attachment background-position background-clip background-origin background-size

- Specifies all background properties in one declaration

```
   background: white url(star.png) no-repeat
top center fixed;
```

- Properties are optional and may appear in any order
- Properties not represented reset to their defaults—be careful it doesn’t overwrite previous background settings.

Multiple Background Images

You can place more than one background image in a single image (separated by commas):

```
  body {
    background:
      url(image1.png) left top no-
repeat,
      url(image2.png) center center no-
repeat,
      url(image3.png) right bottom no-
repeat;
   }
```

Gradient Fills

- A **gradient** is a transition from one color to another.
- **Linear gradients** change colors along a line.
- **Radial gradients** start at a point and spread outward in a circular or elliptical shape.
- You can generate a gradient image for use as a background using **`linear-gradient()`** and **`radial-gradient()`** notation.

Example:

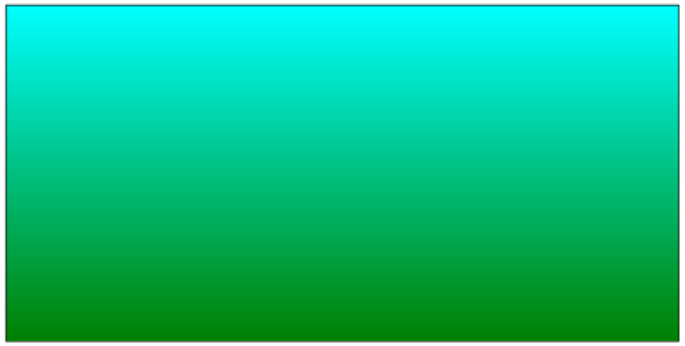
```
 #banner {
   background-image: linear-
gradient(180deg, aqua, green);
 }
```

Linear Gradient

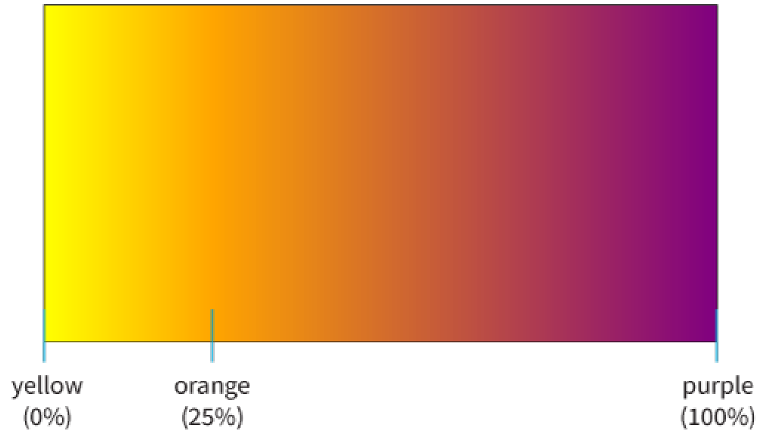
The **`linear-gradient()`** notation provides the angle of a gradient line and the colors the line passes through.

It is specified in degrees (`deg`) or keywords (`to top`, `to right`, `to bottom`, `to left`).

```
linear-gradient(180deg, aqua, green);
or
linear-gradient(to bottom, aqua, green);
```



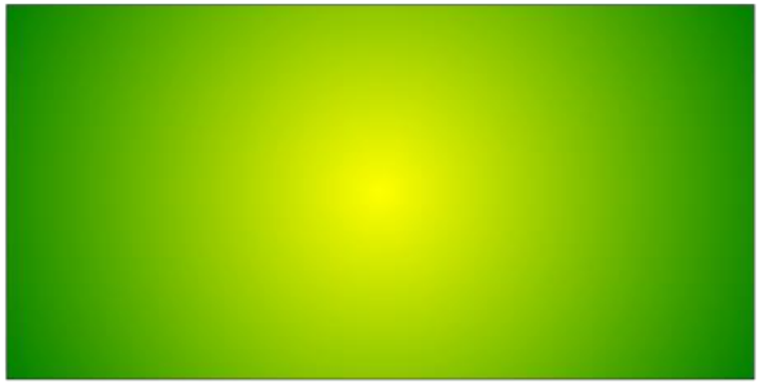
```
linear-gradient(90deg, yellow, orange 25%, purple);
```



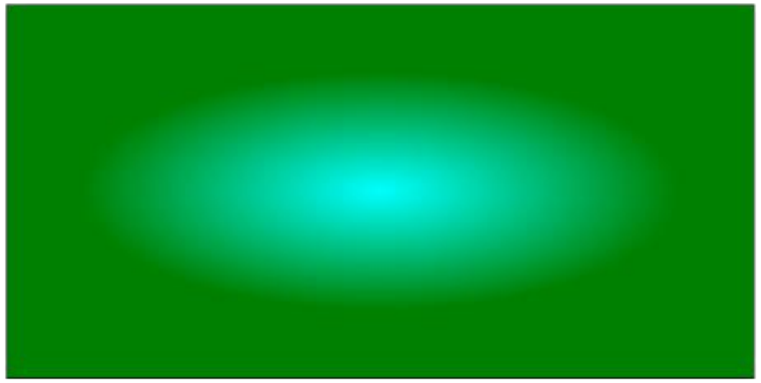
Radial Gradients

The **`radial-gradient()`** notation provides the color values and optional size, shape, and position information.

```
radial-gradient(yellow, green);
```



```
radial-gradient(200px 80px, aqua, green);
```



Gradient Vendor Prefixes

Because the gradient spec has changed over time, gradients require significant prefixing and alternate values:

```
background: #ffff00; /* Old browsers */
background: -moz-linear-gradient(top, #ffff00 0%, #00ff00 100%);
/* FF3.6+ */
background: -webkit-gradient(linear, left top, left bottom, color-
stop(0%,#ffff00), color-stop(100%,#00ff00));
/* Chrome,Safari4+ */
background: -webkit-linear-gradient(top, #ffff00 0%,#00ff00 100%);
/* Chrome10+,Safari5.1+ */
background: -o-linear-gradient(top, #ffff00 0%,#00ff00 100%);
/* Opera 11.10+ */
background: -ms-linear-gradient(top, #ffff00 0%,#00ff00 100%);
/* IE10+ */
background: linear-gradient(to bottom, #ffff00 0%,#00ff00 100%);
/* W3C Standard */
filter: progid:DXImageTransform.Microsoft.gradient( startColorstr='#ffff00',
endColorstr='#00ff00',GradientType=0 );
/* IE6-9 */
```

TIP: Use a tool like Ultimate CSS Gradient Generator: www.colorzilla.com/gradient-editor.

Vendor Prefixes

Browsers once kept experimental implementations of properties separate from the final release by adding a vendor prefix.

Example:

Property name:

```
shape-outside
```

Vendor-prefixed for Safari, Chrome, and Android:

```
-webkit-shape-outside
```

Vendor Prefixes (cont'd)

Prefix	Organization	Most popular browsers
`-ms-`	Microsoft	Internet Explorer
`-moz-`	Mozilla Foundation	Firefox, Camino, SeaMonkey
`-o-`	Opera Software	Opera, Opera Mini, Opera Mobile
`-webkit-`	Originally Apple; now open source	Safari, Chrome, Android, Silk, BlackBerry, WebOS, and many others

NOTE: Browser vendors no longer use the prefix system, but some properties from that era still require them.

ShouldIPrefix.com is a good place to check for properties that still require prefixes.

More Selector Types

Pseudo-class selectors

Pseudo-element selectors

Attribute selectors

Pseudo-Class Selectors

Treat elements in a certain state as belonging to the same class

Link Pseudo-classes

`:link`	Applies style to unvisited (unclicked) links
`:visited`	Applies style to visited links

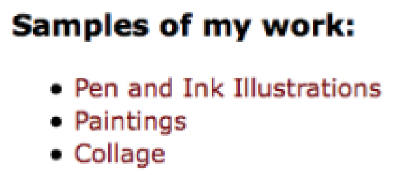
User Action Pseudo-classes

`:focus`	Applies when element is selected for input
`:hover`	Applies when the mouse pointer is over the element
`:active`	Applies when the element (such as a link or button) is in the process of being clicked or tapped

Pseudo-classes (cont'd)

Pseudo-classes must appear in the following order:

```
a { text-decoration: none; }  /* turns underlines off for all links */
a:link { color: maroon; }
a:visited { color: gray; }
a:focus { color: maroon; background-color: #ffd9d9; }
a:hover { color: maroon; background-color: #ffd9d9; }
a:active { color: red; background-color: #ffd9d9; }
```

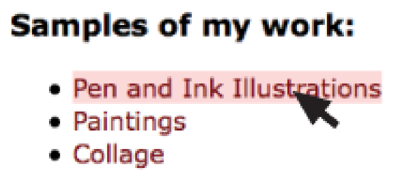


Samples of my work:

- Pen and Ink Illustrations
- Paintings
- Collage

`a:link`

Links are maroon and not underlined.

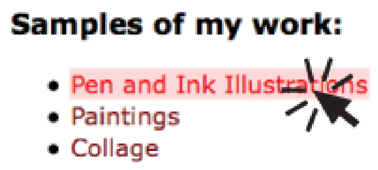


Samples of my work:

- Pen and Ink Illustrations
- Paintings
- Collage

`a:focus`
`a:hover`

While the mouse is over the link or when the link has focus, the pink background color appears.

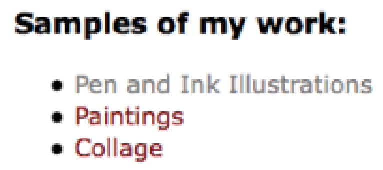


Samples of my work:

- Pen and Ink Illustrations
- Paintings
- Collage

`a:active`

As the mouse button is being pressed, the link turns bright red.



Samples of my work:

- Pen and Ink Illustrations
- Paintings
- Collage

`a:visited`

After that link has been visited, the link is gray.

More Pseudo-Class Selectors

Structural pseudo-classes

These allow selection based on where the element is in the structure of the document (the document tree):

- `:root`
- `:empty`
- `:first-child`
- `:last-child`
- `:only-child`
- `:first-of-type`
- `:last-of-type`
- `:only-of-type`
- `:nth-child()`
- `:nth-last-child()`
- `:nth-of-type()`
- `:nth-last-of-type()`

Input pseudo-classes

These selectors apply to states that are typical for form inputs:

- `:enabled`
- `:disabled`
- `:checked`

Location pseudo-classes (in addition to :link and :visited)

- `:target` (fragment identifier)

Linguistic pseudo-class

- `:lang()`

Logical pseudo-class

- `:not()`

Pseudo-Element Selectors

Applies styles to elements not explicitly marked up in the source.

`::first-line`

Applies a style to the first line of an element:

```
   p:first-line {letter-spacing: 9px;}
```

`::first-letter`

Applies a style to the first letter of an element:

```
 p:first-letter { font-size 300%; color:
                orange;}
```

Pseudo-Element Selectors (cont'd.)

The `::before` and `::after` pseudo-elements insert generated content before or after a specified element.

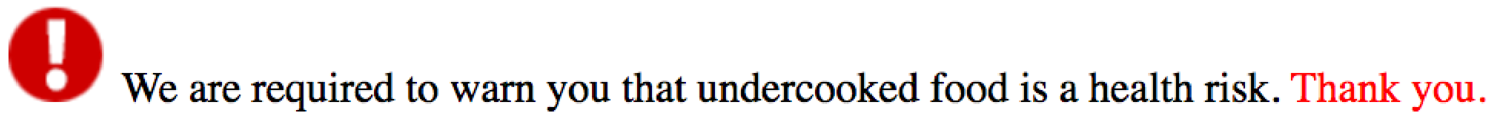
`::before`

Inserts copy (provided with the **`content`** property) before an element and applies style properties to it as specified

`::after`

Inserts copy (provided with the **`content`** property) after an element and applies style properties to it as specified

Generated Content Example



We are required to warn you that undercooked food is a health risk. Thank you.

The style sheet:

```
p.warning::before {
  content: url(exclamation.png);
  margin-right: 6px;
}
p.warning::after {
  content: " Thank you.";
  color: red;
}
```

The markup:

```
<p class="warning">We are required to warn you that
undercooked food is a health risk.</p>
```

Attribute Selectors

Targets elements based on attribute names or values. There are eight types:

Simple attribute selector
Matches an element with a given attribute:

```
E[attribute]
img[title] { border: 3px
solid;}
```

(Matches every `img` *element that has a* `title` *attribute)*

Attribute Selectors (cont’d)

Exact attribute value selector
Matches an element with a specific value for an attribute:

```
E[attribute="exact value"]
img[title="first grade"] {border: 3px
solid;}
```

(matches only if the `title` *value is "first grade")*

Partial attribute value selector (~)
Matches an element by one part of an attribute value.:

```
E[attribute~="value"]
img[title~="grade"] {border: 3px solid;}
```

(matches "first grade", "second grade", and so on)

Attribute Selectors (cont'd.)

Hyphen-separated attribute value selector (|)
Targets hyphen-separated values:

```
E[attribute|="value"]
```

Beginning substring attribute value selector (^)
Matches an element with attribute values that start with the given string of characters:

```
E[attribute^="first part of a value"]
```

Ending substring attribute value selector ($)
Matches an element with attribute values that end with the given string of characters:

```
E[attribute$="last part of a value"]
```

Arbitrary substring attribute value selector (*)
Looks for the text string in any part of the attribute value:

```
E[attribute*="any part of the value"]
```

External Style Sheets

- Store styles in a separate *.css* file and attach to the document via **`<link>`** or **`@import`**
- Most efficient method: Change styles in multiple documents by editing one *.css* file
- A *.css* document is a simple text document (may begin with **`@charset`** to identify character set)

Attaching a Style Sheet with the link Element

- The **`link`** element defines a relationship between the current document and an external resource.
- It goes in the **`head`** of the document.
- Use the **`rel`** attribute to say it's a style sheet. Use **`href`** to provide the URL of the *.css* file (relative to the current document):

```
<head>
  <title>Titles are required.</title>
  <link rel="stylesheet" href="/path/stylesheet.css">
</head>
```

Attaching a Style Sheet with an @import rule

An **`@import`** rule imports the contents of an external style sheet into another style sheet (either a *.css* document or embedded with **`style`**):

```
<head>
  <style>
     @import
url("/path/stylesheet.css");
     p { font-face: Verdana;}
  </style>
  <title>Titles                           are
required.</title>
</head>
```