HTML & CSS: LEVEL 1

Instructor: Beck Johnson

Week 2

SESSION OVERVIEW

- Week 1 review and questions
- Image format overview
- Optimizing web images
- Introduction to CSS

REVIEW!

REVIEW: WEBPAGE COMPONENTS

- **HTML** structures and organizes content
- **CSS** stylizes the content and creates layout
- **Javascript** adds interactivity

REVIEW: HTML DOCUMENTS

- `<!DOCTYPE html>` tells the browser it's serving an HTML file using HTML5 standards
- `<html>` wraps the whole document
- `<head>` wraps the metadata
- `<body>` wraps the visible content
- Most HTML elements have **opening** and **closing tags**, and some have **attributes**

REVIEW: HTML CONTENT

- **Headings** create an header/outline

  `<h1>...<h6>`

- **Paragraphs** and **lists** structure text

  `<p>`
  `<ul>`
  `<ol>`

- **Images** and **links** both require **attributes** to work

IMAGES

```
<img src="kitten.jpg" alt="Cute kitten" />
```

- Does not have a closing tag
- Two required **attributes**:
  - **src** is where the file lives (local or external)
  - **alt** is a description of the image (used for screen readers, search engines, etc)

LINKS

```
<a href="http://google.com">Google</a>
```

- Creates a link to other pages or websites
- The `href` attribute says where the link should go
- Anything inside `<a>` tags is clickable

QUESTIONS?

WEB IMAGES

WEB-READY IMAGES

- **Minimize** file sizes to help load times in browser
- **Optimizes** images for RGB displays with correct **resolution** for browsers
- **Flattens** layers and removes metadata from graphics

WEB IMAGE TYPES

JPG/JPEG

- Created by the “Joint Photographic Experts Group”
- Millions of colors
- Uses a compression algorithm called **lossy**

GIF

- Stands for “Graphics Interchange Format”
- 256 colors max - fewer colors mean a smaller file
- Animation and off-on transparency

PNG

- Stands for "Portable Network Graphics”
- Millions of colors
- Full alpha transparency

JPGS

JPG PROS:

- small file size
- rich colors

JPG CONS:

- image distortion
- no transparency

JPGS

GIFS

GIF PROS:

- small file size
- transparency
- animations

GIF CONS:

- few colors

PNGS

PNG PROS:

- any amount of transparency
- best image quality

PNG CONS:

- large file size
- IE 7&8 don't support transparency

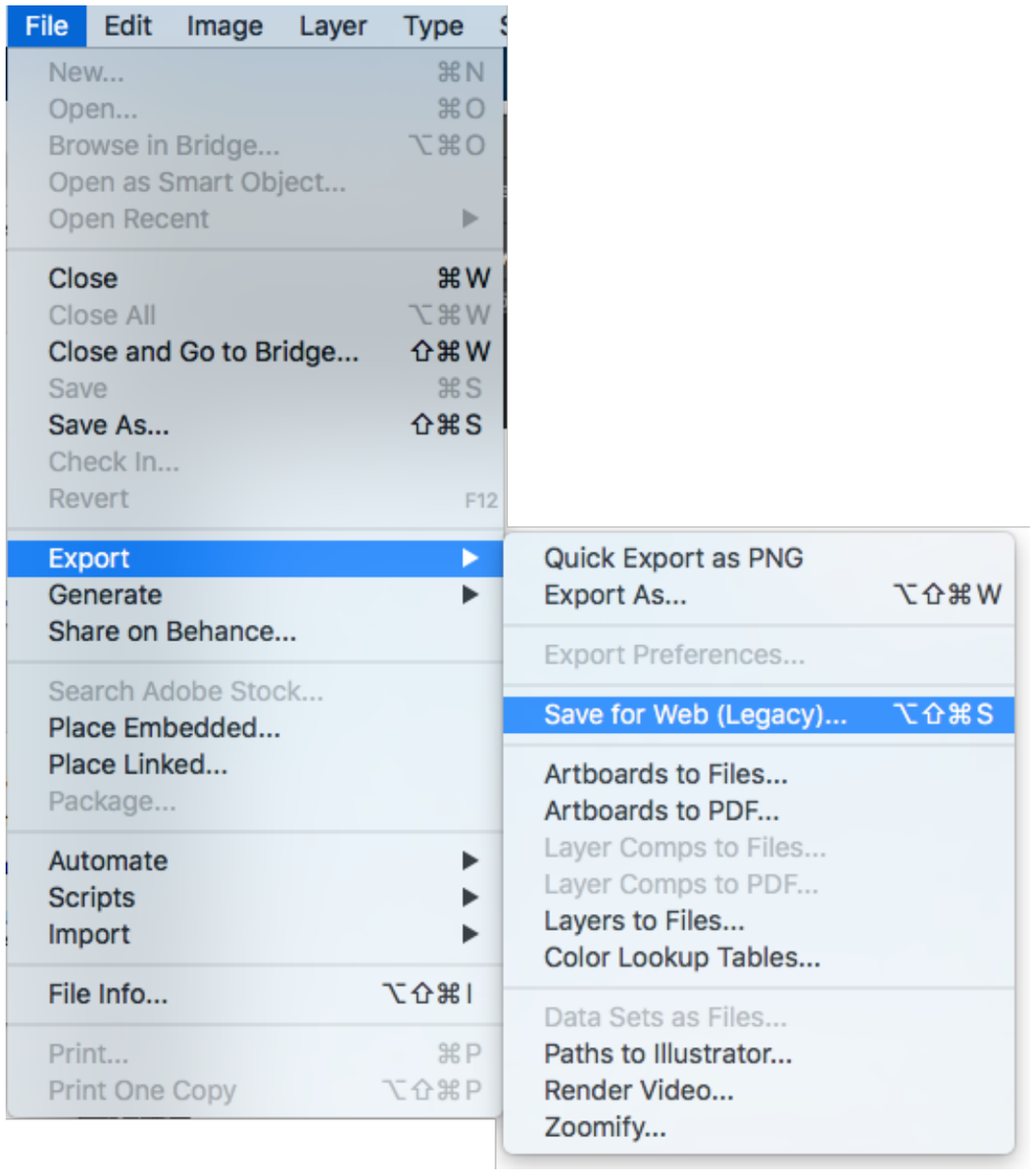
BEST USES:

WEB IMAGES SUCH AS LOGOS THAT INVOLVE TRANSPARENCY AND FADING.

IMAGES IN THE MIDDLE OF THE EDITING PROCESS.

COMPLEX IMAGES LIKE PHOTOGRAPHS IF FILE SIZE IS NOT AN ISSUE.

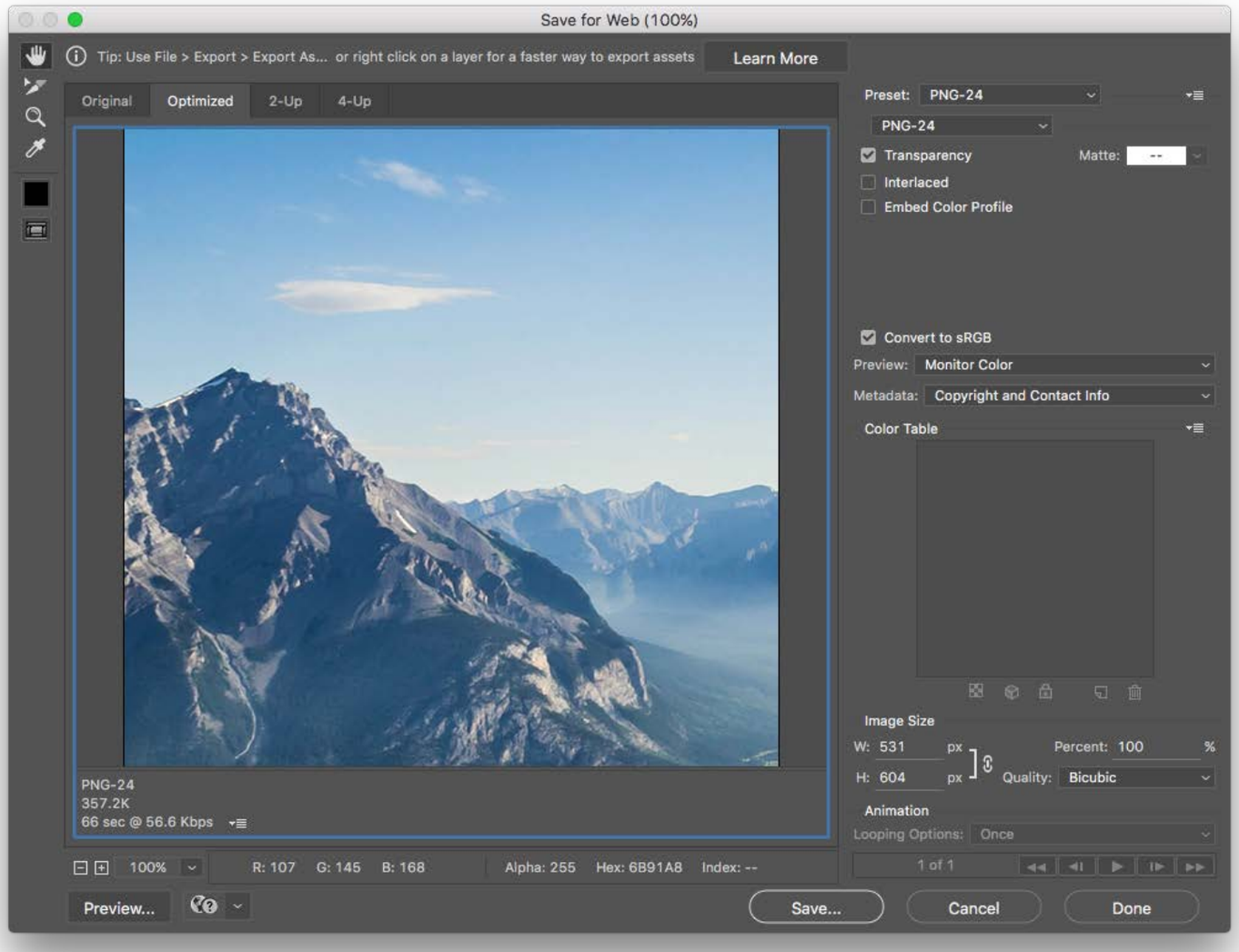
“SAVE FOR WEB” IN ADOBE CS



Adobe products have a “**Save for Web...**” or “**Save for Web and Devices...**” option

“SAVE FOR WEB” IN ADOBE CS

- Click **File > Export > Save for Web... (**or **Export As)**
- Choose a format **(JPEG, PNG 24, or GIF)**
- Adjust image size to max size display
- Save to your images directory



SOME “GOTCHAS”

- Best practice to work in 72 PPI in graphic editor programs. (keeps file sizes down)
- Always work in **RGB** when working with graphics for the web. **CMYK** is for print
- Graphics for **Retina devices** need to be saved out at 2X their “normal” size

CASCADING STYLE SHEETS

- CSS is a language for specifying how documents are presented to users
- Ability to override the browser's default presentation styles with custom versions
- Provides consistent and scalable ways to style single elements, single pages, or entire websites
- Separates look and feel from content/markup

CASCADING STYLE SHEETS: FAIR WARNING

- There is **A LOT** you can do with CSS
- We won't get anywhere close to covering everything!
- We will cover CSS for text styles, colors, positioning, layout, and a couple of extras

WHY USE CSS?

- Helps you avoid duplication by keeping styles in one place (one external stylesheet)
- Makes style maintenance easier - for example, update the font for the whole site in one line of code!
- Separating presentation from content enforces style consistency and allows flexibility

ANATOMY OF A CSS RULE

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

- `selector` is the **thing** you want to style
- `property` is the **attribute** you want to style
- `value` is how you want to style it
- Values always end in semicolons (`;`)

ANATOMY OF A CSS RULE

So!

```
p { color: blue; }
```

"All paragraphs will have blue text "

EXAMPLE CSS RULE

```
p { color: blue; }
```

- `selector` is `p` (all `<p>` tags in the HTML)
- `property` is `color`
- `value` is `blue` (many color names are supported, or use the hex code `#0000ff`)

EXAMPLE CSS RULE

```
p {
    color: blue;
    font-size: 14px;
}
```

- Multiple `properties` can be defined for a single `selector`, each separated by a semi-colon (`;`)

{} COMMON FONT PROPERTIES

`font-style:` normal by default - can also be *italic* or *oblique*

`font-weight:` normal by default - can also be **bold**, or values of 100, 200, etc (depending on the typeface)

`font-family:` the name of a typeface installed on the user's computer

```
p {
    font-family: Arial, Helvetica, sans-serif;
}
```

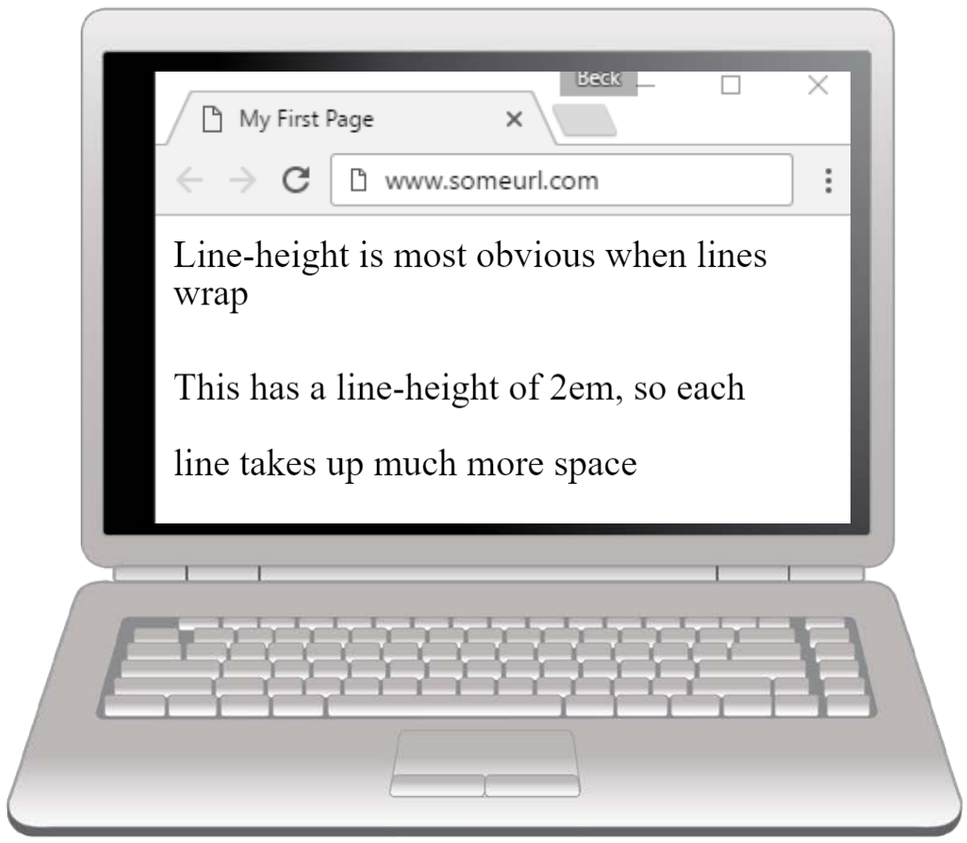
- The W3 has a list of "web safe" fonts that most people will have installed locally

{} COMMON FONT PROPERTIES

`line-height:` a number followed by a measurement of the height of a line of that element, in ems (em) or pixels (px)

- similar to **leading** in typography

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



`font-size:` a number followed by a measurement of the height of that element's text in ems (em) or pixels (px)

```
p { font-size: 14px; }
```

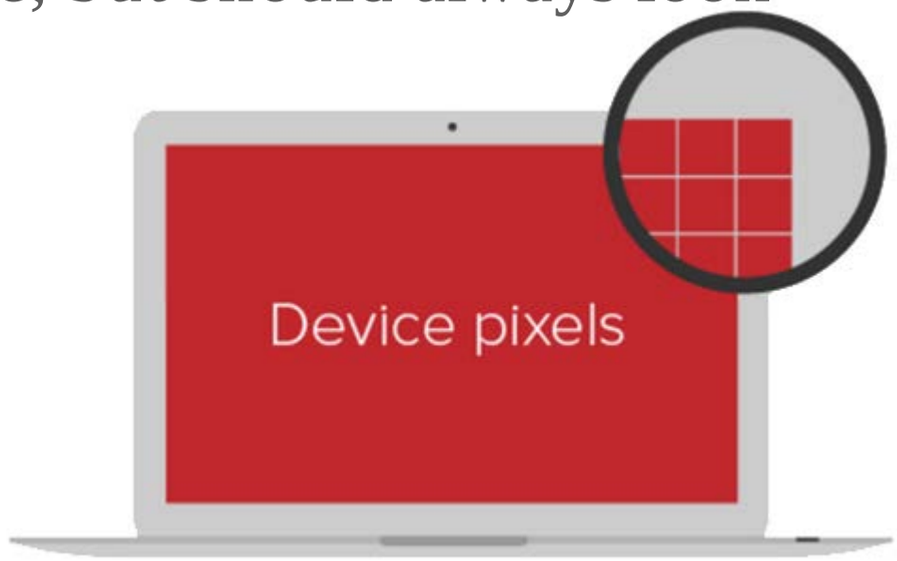
{} QUICK ASIDE ABOUT UNITS

The two standard units for sizing in CSS are **px** and **em**

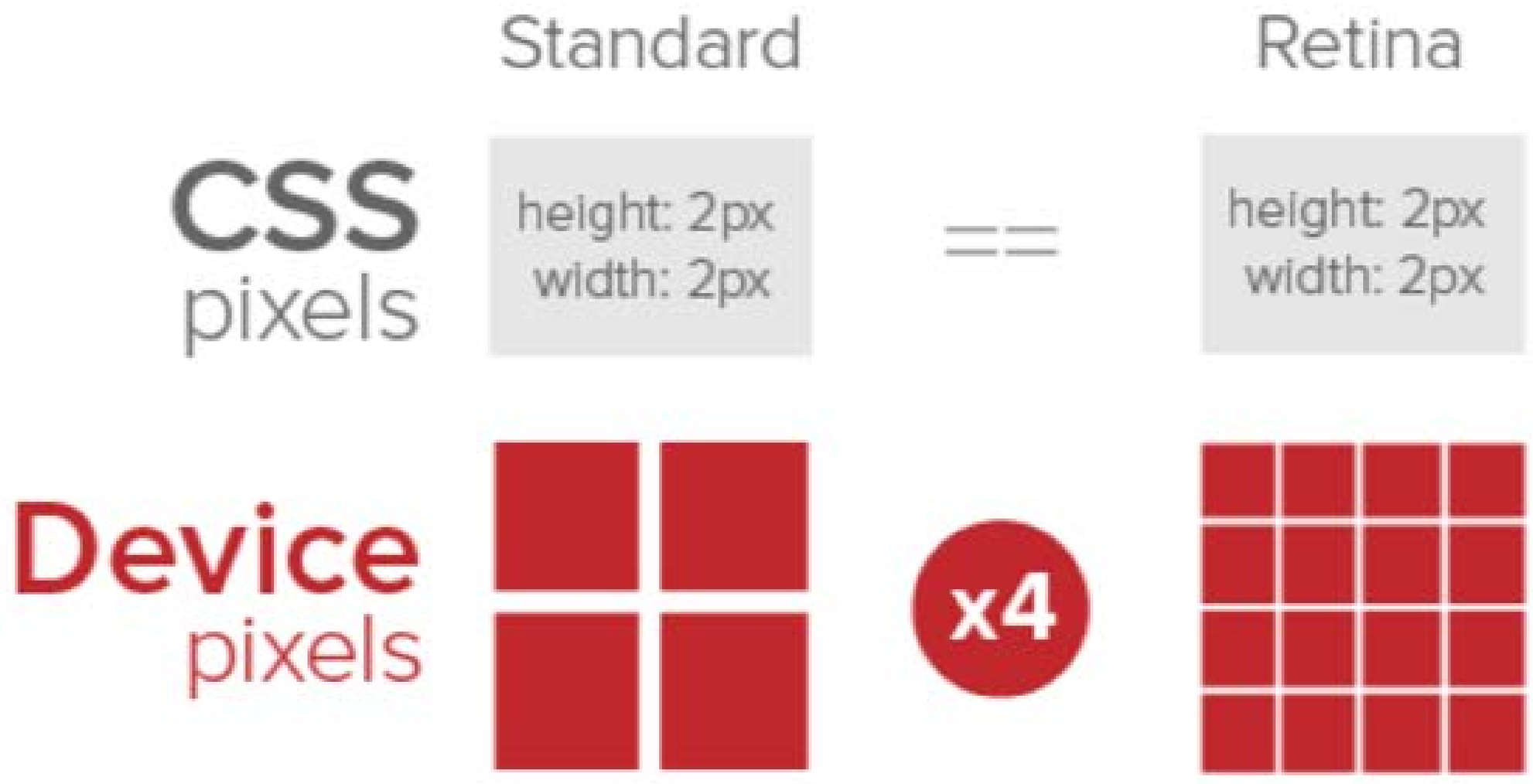
- **px** is an abstract unit that isn't related to font height and isn't a physical unit of measurement

- Devices with more PPI (pixels per inch) may use several "device" pixels when displaying a 1px line

- That means that px size varies by device, but should always look "about the same"



{} QUICK ASIDE ABOUT UNITS



{} AH-EM

- **em** refers to the height of the letter 'm' of the font being used
    - This unit of measurement is a description of the **relative** size between this element and its parent
    - So `h2 { font-size: 2em; }` means the header is 2 times as big as the letter 'm' of the default font in your html document

{} THAT WASN'T QUICK

Because em is relative, that means that if the parent's font size is increased, the children will get bigger too.

	`body { font-size: 100%; }`	`body { font-size: 120%; }`
`font-size: 1em`	The quick brown fox	The quick brown
`font-size: 12px`	The quick brown fox	The quick brown fox

{} COLORS

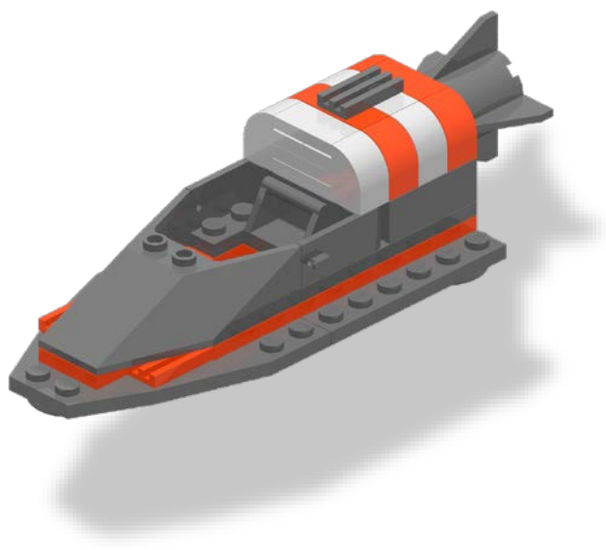
- `color:` changes the color of text
- `background-color:` sets the background color of an element
- Color `value` can be set using **names**, **HEX, RGB,** or **RGBA**
  - Name: `white`
  - Hex: `#ffffff`
  - RGB: `rgb(255, 255, 255)`
  - RGBA: `rgba(255, 255, 255, 0.8)`

{} FOUR LINK STATES

You can style a link differently depending on what **state** it's in

a:link

a:visited



a:hover

a:active

{} FOUR LINK STATES

```
a { color: blue; }

a:visited { color: gray; }

a:hover { color: purple; }

a:active { color: yellow; }
```

Let's inspect a live demo of how this looks

{} TEXT-ALIGN

You can change the alignment of text using the `text-align` property.

Values:

- `center`
- `left`
- `right`
- `justify`

```
h1 { text-align: center; }
```

{} MULTIPLE SELECTORS & PROPERTIES

- You can add multiple `selectors` to a CSS rule
- You can add multiple `properties` to a CSS rule
- Example: style all ordered and unordered lists:

```
ul,
ol {
    font-size: 16px;
    font-weight: bold;
    color: #444444;
}
```

PRACTICE TIME!

PRACTICE

- Open the website we created last week.
- Add a `<style></style>` section in the `<head>` on your homepage (index.html)
- Make some style changes using CSS
    - Consider changing font color, font family, font size, link color, text alignment, and background colors

{} CSS IN MULTIPLE PLACES

- **Inline styles** are applied to only a single element

- **Internal styles** are added in the <head> of a page and style only that page

- **External stylesheets** are called into multiple pages, and are declared in separate .css files

{} THE "CASCADING" PART

The beauty of CSS is being able to create styles and then override them when you want to customize the look of your pages.

There are **3 rules** for determining how styles get applied:

- Styles are applied from far to near
- Styles are applied from top to bottom
- Children elements are more specific than parents

{} STYLES "LOCATION"

Styles that are "closer" to the elements they style take precedence.

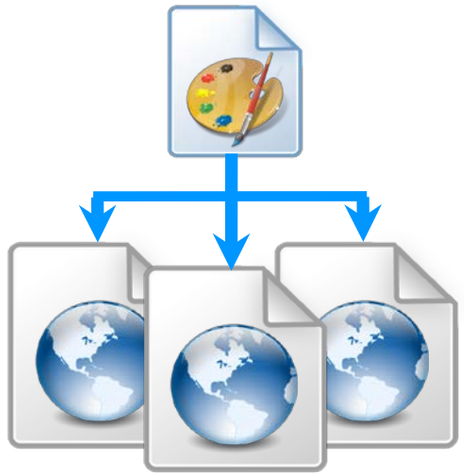
- Browser default
- External styles (in a **.css** file)
- Internal styles (in the <head>)
- Inline styles (directly on an element)

Less Specific

Most Specific

{} STYLES “LOCATION”

Browser default



External styles
(in a **.css** file)

Internal styles
(in the <head>)

Inline styles
(directly on an element)

{} TOP TO BOTTOM

If the same property is styled multiple times for the same selector, **the last one wins**

```
p { color: #2f4251; }

p { color: #daa645; } /* this one wins */
```

CSS COMMENTS

Just like HTML, CSS can have **comments.**

```
<style>
  /* I am a CSS comment! */

  h1 { /* I am also a CSS comment */
    color: #ff0000;
  }
</style>
```

{} CHILDREN ARE SPECIFIC

Children elements usually **inherit** styles from their parents but can **override** parents with their own styles

```
body { color: #2f4251; } /* parent */

p { color: #daa645; } /* child */
```

All text in the body that is NOT a paragraph will be dark gray. Paragraphs will be mustard-colored

{} SELECTORS CAN BE MORE SPECIFIC

If one style is **more specific** than another, it takes precedence

```
p { color: #daa645; } /* all paragraphs */

a { color: #e7c0c8; } /* links in general */

p a { color: #c4fe46; } /* links in paragraphs */
```

{} PRACTICE (AGAIN!)

{} EXTERNAL STYLESHEETS

- **Copy and paste** your styles from inside `<style></style>` in index.html into a new file.

- Save your new files as **styles.css,** and save it in your **css** directory/folder.

- **Remove** the `<style></style>` tags from index.html

{} LINKING TO EXTERNAL STYLESHEET

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

- Tells the browser to find and load the styles.css file from the css directory
- The `rel` attribute stands for "relation" - in this case, this link's relationship to the document is "stylesheet"
- This tag goes inside the `<head>` element
- Should be on every page that needs the styles

CSS PSEUDO CLASSES

{} ANCHOR PSEUDO CLASSES

A **CSS pseudo-class selector** specifies a special state of the element to style

We already saw this used for links:

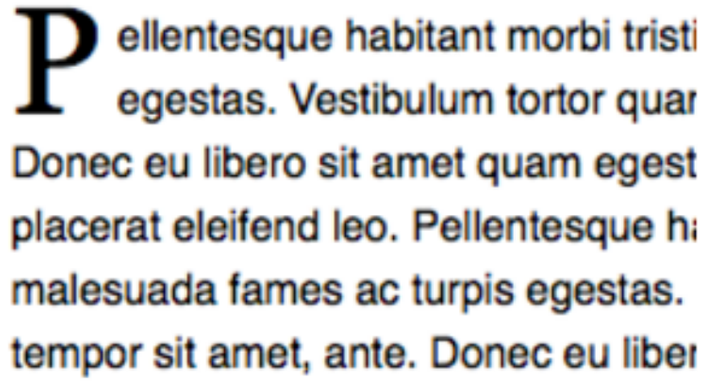
```
a:hover { color: purple; }
```

{} OTHER PSEUDO CLASSES

`:first-letter` styles the first letter of a block of text

`:first-child` and `:last-child` style the first and last children of a parent

```
p:first-child:first-letter {
      font-size: 3em;
      float: left;
}
```



{} OTHER PSEUDO CLASSES

`:focus` styles an element that has the current keyboard focus, from either click or tab

`:checked` styles a selected radio button or checkbox

PRACTICE TIME!

HOMEWORK

- Using what we learned about image formats, decide if the images on your website are in the best format

- Re-save images in an optimized format using Photoshop

- Style at least one pseudo-class

“HOMEWORK”

- Practice!

- Optional: read chapters 10-12 and chapter 16 of HTML and CSS: Design and Build Websites

- Check out the CSS Zen Garden for inspiration on how simply changing CSS can change the entire look and feel of a page