Lesson Monday

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / HTML Block Elements Part 2

Text Cheat sheet

Commands

- `pwd` command in terminal returns the directory currently located in.

```
$ pwd
/Users/staff/Desktop/my-first-webpage
```

Example

favorite-things.html

```
<!DOCTYPE html>
<html lang="en-US">
<head>
  <title>Michael's favorite things</title>
</head>
<body>
  <h1>My favorite things</h1>
  <h2>These are a few of my favorite things.</h2>

  <h3>People</h3>
  <p>Here are some of my favorite people!</p>
  <ul>
    <li>My brother Christopher</li>
    <li>My mom and dad:
      <ul>
        <li>Steve</li>
        <li>Carol</li>
      </ul>
    </li>
    <li>My friend Jessica</li>
    <li>My friend Candy</li>
  </ul>

  <h3>Flavors of ice cream</h3>
  <p>These are my three favorites, in order:</p>
  <ol>
    <li>Pistachio</li>
    <li>Malted milk chocolate</li>
    <li>Black licorice (really!)</li>
  </ol>

  <p>Thanks for reading about me!</p>
</body>
</html>
```

Previous (/introduction-to-programming/git-html-and-css/html-block-elements-p-tags-headings-list-elements-and-more)
Next (/introduction-to-programming/git-html-and-css/practice-block-elements)

Lesson 17 of 64
Last updated March 24, 2023

disable dark mode

Epicodus (http://www.epicodus.com)

Lesson Tuesday

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / HTML Inline Elements and Attributes: <strong>, <em>, <img>, <a>, and More

Text Cheat sheet

Terminology

- **Inline element**: HTML elements that do not appear on their own line, but instead share a line with other elements.
- **Attribute**: Additional information provided to an HTML tag. For instance, the `href` attribute in an `<a>` tag provides the URL a link should travel to.
- **Relative path**: A path to a file within the project itself. Usually referring to the location provided in as the `href` attribute in an `<a>` tag.
- **Self-closing**: An HTML element that does not require a closing tag, such as `<img>`.

Inline Elements

Some inline elements:

- `<strong>` : Makes text appear bolder. Example: `<strong>` **This content is important.** `</strong>`
- `<em>` : Emphasizes text. Similar to italics. Example: `<em>` *This content is emphasized.* `</em>`
- `<a>` : Anchor tag. Creates a link. The URL following `href` denotes where the link should travel to. Example: `<a href="http://www.epicodus.com">Epicodus</a>` — link to another web page
- `<img>` : Image tag. `<img src="img/kitten.jpeg" alt="A photo of a cute kitten.">` — link to an image located in `img` subdirectory, with alt text

Previous (/introduction-to-programming/git-html-and-css/commit-trailers-and-github-contributions)
Next (/introduction-to-programming/git-html-and-css/practice-inline-elements)

Lesson 22 of 64
Last updated March 24, 2023

disable dark mode

(http://www.epicodus.com)

Lesson Tuesday

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / CSS: Styling Text and Best Practices

Text Cheat sheet

Terminology

- **CSS**: Stands for Cascading Style Sheets. A language used to program the visual appearance of HTML elements.
- **Rule**: A block of CSS that details particular stylistic instructions to be applied to an HTML element.
- **Selector**: The part of a CSS rule that determines which HTML elements the rule applies to.
- **Declaration Block**: Designated by curly brackets `{ }`, this is where we define the CSS styles.
- **Property**: The characteristic a CSS rule is altering. (For example, `color`, or `font-size`).
- **Value**: The attribute a CSS rule is applying to the specified property.
- **Declaration**: The combination of a property and value together, for example `color: blue`; every declaration block can have multiple declarations separated by a semicolon `;`.

Inline Styles Examples

- **Inline Styles**: CSS styles listed directly within the HTML body. To add inline styles, you need to add a `style` attribute to the element that you wish to style.
  - **Note:** We won't use inline styles frequently.

```
<h1 style="color:blue;background-color:hotpink">My favorite things</h1>
```

External Stylesheet Examples

- **External Stylesheet**: CSS styles listed in a separate `.css` file.
- Even though we can set inline styles for our HTML elements using the `style` attribute, it's best practice to list CSS styles in a separate file (with a `.css` extension) for these reasons:
  - We can set styles that apply to multiple elements.
  - When we separate our code into multiple files, it's easier to read and update.

Linking to an External Stylesheet from HTML Document

To tell an HTML document to use a `css/styles.css` file for your website's CSS styles, add a new `<link>` tag to the `<head>` tags of our document:

favorite-things.html

```
<!DOCTYPE html>
<html lang="en-US">
<head>
  <link href="css/styles.css" rel="stylesheet" type="text/css">
  <title>Michael's favorite things</title>
</head>
<body>
  ...
</body>
</html>
```

HTML Link elements direct our HTML document to load resources stored in separate files:

- To specify where the resource is located, we give a value to the `href` attribute that contains the path to our file. Our `href` attribute contains `css/styles.css` as the value, because *in relation to this favorite-things.html file*, the `styles.css` file is in a subdirectory named `css`.
  - If the `styles.css` file were in the same directory as the favorite-things.html file, then the link would just be to `"styles.css"` rather than to `"css/styles.css"`.
- To describe what type of resource we're loading we specify two additional attributes:
  - `rel` specifies the relationship of the resource to our HTML document. In this case, the external resource is a `stylesheet`!
  - `type` specifies the media type of the resource. We list `text/css` because our resource is a CSS stylesheet.

Example CSS Rules in an External Stylesheet

styles.css

```
h1 {
  color: #0000ff;
  text-align: center;
}

h2 {
  font-style: italic;
  text-align: center;
}

p {
  font-family: sans-serif;
}

ul {
  font-size: 20px;
  line-height: 30px;
}
```

In the code above...

- Each individual block is a **rule**.
- `h1`, `h2`, `p`, and `ul` are all **selectors**.
- `color`, `text-align`, `font-style`, `font-family`, `font-size` and `line-height` are all **properties**.
- `blue`, `center`, `italic`, `sans-serif`, `20px` and `30px` are all **values**.
- The entire file `styles.css` is a **stylesheet**.

Additional Resources

- W3School's CSS Reference (https://www.w3schools.com/css/default.asp)
- MDN Web Doc's CSS Reference (https://developer.mozilla.org/en-US/docs/Web/CSS)
- CSS Zen Garden (http://csszengarden.com/)

Previous (/introduction-to-programming/git-html-and-css/practice-inline-elements)
Next (/introduction-to-programming/git-html-and-css/debugging-html-and-css)

Lesson 24 of 64
Last updated March 24, 2023

disable dark mode

Epicodus (http://www.epicodus.com)

Lesson Tuesday

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / Debugging HTML and CSS

Text Cheat sheet

Terminology

- **DevTools** or **Developer Tools**: An area of the Google Chrome web browser that offers debugging options, and other tools helpful to web developers to assist in locating problems, measuring performance, etc.
- **JavaScript Console**: Location in Google Chrome where a website's errors are logged, and other debugging tools are available.
- **Inspecting an element**: The act of looking at the code for a specific element on the page.

How To

- **Open JavaScript Console in Chrome:** In Chrome, click on the menu button (it looks like ☰ on the right side of address bar), go to the *More Tools* , and click *Developer Tools*. You may also have to click the *Console* tab at the top of the window that pops open.

- **Keyboard shortcut for JavaScript Console in Chrome:** CMD + Option + J
- **How to inspect an element**: Right click on an element of a web page, and select "Inspect" from the resulting menu of options.

Previous (/introduction-to-programming/git-html-and-css/css-styling-text-and-best-practices)
Next (/introduction-to-programming/git-html-and-css/practice-styling-text)

Lesson 25 of 64
Last updated March 24, 2023

disable dark mode

(http://www.epicodus.com)

Lesson Tuesday

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / Git Branching

Text Cheat sheet

Terminology

- **Branching**: The act of making multiple copies of the same code in the same project. Usually in order to do different things with the same code base, or to allow multiple people to work on the same code at the same time.
- **Branch**: Another copy of your code
- **Main Branch**: Like the "final draft" of your code.
- **Feature Branches**: A common term for branches that are *not* `main`. Usually branches meant for experimenting with new features before adding them to the final version of the project.

Tips

- The `$ git branch` command with nothing after it will display all branches in a project. The branch you're currently located in will be denoted with an asterisk `*`.

- The `$ git branch` command with a branch name after it (ie: `$ git branch <name-of-new-branch>`) will create a new branch of the name you specify. Branch names should be short and meaningful, and describe the reason for the branch.
- We can switch which branch we're viewing by using the `$ git checkout` command with the name of the branch we'd like to see. For instance, `$ git checkout blue_theme` will move us to a branch named `blue_theme`.

Previous (/introduction-to-programming/git-html-and-css/practice-styling-text)

Next (/introduction-to-programming/git-html-and-css/git-merging)

Lesson 27 of 64

Last updated March 24, 2023

disable dark mode

Epicodus (http://www.epicodus.com)

Lesson Tuesday

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / Git Merging

Text Cheat sheet

Terminology

- **Merging**: The act of combining two different branches together.

Overview

- To merge one branch into another, you must first navigate to the branch you'd like to introduce new code into.
- Once located in that branch, the command `$ git merge <branch-name>` will merge the commits from the outside branch into the current branch.
- If the two branches don't contain edits to the same areas, Git will be able to merge these two branches automatically.
- If two branches contain edits to the same areas of code, this will result in a merge conflict, and Git will request you resolve it manually. We'll explore merge conflicts in detail later on.

Previous (/introduction-to-programming/git-html-and-css/git-branching)
Next (/introduction-to-programming/git-html-and-css/practice-branching-and-merging)

Lesson 28 of 64
Last updated March 24, 2023

disable dark mode

(http://www.epicodus.com)

Lesson Tuesday

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / Markdown

Text Cheat sheet

Headings

Headings are designated with a `#`. Text with 1 `#` is the largest and text with 6 `#`s is the smallest.

```
# Best Chocolate Chip Cookies
## Best Chocolate Chip Cookies
### Best Chocolate Chip Cookies
#### Best Chocolate Chip Cookies
##### Best Chocolate Chip Cookies
###### Best Chocolate Chip Cookies
```

becomes

Best Chocolate Chip Cookies

Best Chocolate Chip Cookies

Best Chocolate Chip Cookies

Best Chocolate Chip Cookies

Best Chocolate Chip Cookies

Best Chocolate Chip Cookies

Italics

To create italics, surround the text with single underscores `_`.

Example:
`_Be warned_, these will fly off of the plate!`

becomes:

Be warned, these will fly off the plate!

Bold

To create a bold text, surround the text with double asterisks `**.

Example: `Bake for **10 minutes** or until golden brown.`

becomes:

Bake for **10 minutes** or until golden brown.

Ordered lists (numbered)

Ordered list just use the number and period for each line.

```
1. Beat the butter...
2. Mix the flour...
```

Unordered lists (bullet points)

Bullet points are made with the `*` and a single space. There must also be a blank line before the first item on the list.

Example:

```
Ingredients

* butter
* flour
* brown sugar
* maple syrup
```

becomes:

Ingredients

- butter
- flour
- brown sugar
- maple syrup

Links

Two elements make up a Markdown link, the text that will display on the page surrounded by `[]` and the web address where the link will go surrounded by `()`.

Example: `[Click here](http://allrecipes.com/) to check out my other great recipes.`

Images

Like links except the square brackets are preceded by `!` and contain *alt text* for the visually impaired, and the `()` can contain a web address for an image.

Example: `![An image of a cookie](http://lorempixel.com/400/200/)`

Resources

For more on Markdown: Daring Fireball (http://daringfireball.net/projects/markdown/syntax).

Previous (/introduction-to-programming/git-html-and-css/practice-branching-and-merging)
Next (/introduction-to-programming/git-html-and-css/writing-a-readme)

Lesson 30 of 64
Last updated February 28, 2023

disable dark mode

Epicodus (http://www.epicodus.com)

Lesson Tuesday

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / Writing a README

Text Cheat sheet

Terminology

- **README**: A file located in a project's repository that provides detailed information about a project.
- **Markdown**: A simple language that uses marking annotation to format text for display in a browser.
- **Open source**: Software that is freely distributed by its creator(s) for use and distribution.

Examples

Headings

Headings are designated with a `#`. Text with 1 `#` is the largest and text with 6 `#`s is the smallest.

```
# Best Chocolate Chip Cookies
## Best Chocolate Chip Cookies
### Best Chocolate Chip Cookies
#### Best Chocolate Chip Cookies
##### Best Chocolate Chip Cookies
###### Best Chocolate Chip Cookies
```

becomes

Best Chocolate Chip Cookies

Best Chocolate Chip Cookies

Best Chocolate Chip Cookies

Best Chocolate Chip Cookies

Best Chocolate Chip Cookies

Best Chocolate Chip Cookies

Italics

To create italics, surround the text with single underscores `_`.

Example:
`_Be warned_, these will fly off of the plate!`

becomes:

Be warned, these will fly off the plate!

Bold

To create a bold text, surround the text with double asterisks `**`.

Example: `Bake for **10 minutes** or until golden brown.`

becomes:

Bake for **10 minutes** or until golden brown.

Ordered lists (numbered)

Ordered list just use the number and period for each line.

```
1. Beat the butter...
2. Mix the flour...
```

Unordered lists (bullet points)

Bullet points are made with the `*` and a single space. There must also be a blank line before the first item on the list.

Example:

```
Ingredients

* butter
* flour
* brown sugar
* maple syrup
```

becomes:

Ingredients

- butter
- flour
- brown sugar
- maple syrup

Links

Two elements make up a Markdown link, the text that will display on the page surrounded by `[]` and the web address where the link will go surrounded by `()`.

Example: `[Click here](http://allrecipes.com/) to check out my other great recipes.`

Images

Like links except the square brackets are preceded by `!` and contain *alt text* for the visually impaired, and the `()` can contain a web address for an image.

Example: `![An image of a cookie](http://lorempixel.com/400/200/)`

Additional Resources

For more on Markdown: Daring Fireball (http://daringfireball.net/projects/markdown/syntax).

Previous (/introduction-to-programming/git-html-and-css/markdown)
Next (/introduction-to-programming/git-html-and-css/practice-markdown-by-writing-readmes)

Lesson 31 of 64

Last updated March 24, 2023

disable dark mode

Epicodus (http://www.epicodus.com)

Lesson Tuesday

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / GitHub Pages

Text Cheat sheet

Using GitHub Pages

After you've finished your project, you can create a `gh-pages` branch either locally or on GitHub to allow your project to be viewable at *my-github-username.github.io/repository-name*

Create Branch Locally

To create a `gh-pages` branch locally and then push to GitHub:

```
$ git branch gh-pages
$ git checkout gh-pages
$ git push [remote nickname] gh-pages
```

Create Branch on GitHub

To create a `gh-pages` branch from GitHub:

1. Click the branch drop down on the left (when viewing the repository on GitHub)
2. Type `gh-pages` ,
3. Select `Create branch: gh-pages`

Previous (/introduction-to-programming/git-html-and-css/practice-markdown-by-writing-readmes)
Next (/introduction-to-programming/git-html-and-css/practice-github-pages)

Lesson 33 of 64

Last updated March 25, 2023

disable dark mode

Lesson Wednesday

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / Styling with Classes and Ids

Text Cheat sheet

Terminology

- **Lorem ipsum**: Text that is used as filler to display or demonstrate an area of text; it looks like Latin and allows the user to focus on the overall layout without focusing on the meaning of words. Fun fact: "lorem ipsum" derives from the Latin for "pain itself".
- `class` **attribute**: Does not have to be unique and can be applied to multiple elements. Classes are used to group one or more elements.
    - When referencing **classes** in CSS, use: `.` (e.g. `.intro`)
- `id` **attribute**: Must be unique and can only be applied to one element. Ids are used to distinguish one element from the rest.
    - When references **ids** in CSS, use: `#` (e.g. `#line2`)

Examples

webpage-with-classes.html

We can add one or more classes to one or more elements, but we can only add one id to an element and it must be unique in the webpage.

```
<p class="important emphasize">Important stuff should be:</
p>
<ul class="important">
  <li>red,</li>
  <li>bold, and</li>
  <li id="large">big, but only for paragraphs.</li>
</ul>
```

Create CSS rules for that class:

styles-for-classes.css

```
.important {
  color: red;
}

.emphasize {
  font-weight: bold;
}

#large {
  font-size: 40px;
}
```

To create a CSS rule for only a specific a tag with a class:

styles-for-classes.css

```
p.important {
  font-size: 24px;
}
```

Previous (/introduction-to-programming/git-html-and-css/journal-1-discussion)
Next (/introduction-to-programming/git-html-and-css/practice-styling-with-classes-and-ids)

Lesson 39 of 64
Last updated March 24, 2023

disable dark mode

Epicodus (http://www.epicodus.com)

Lesson Wednesday

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / HTML Divs and Spans, and CSS Sizing Units

Text Cheat sheet

Examples

Group elements together with similar styles using `<div>` s. For example, change this:

stuff.html

```
<h2 class="important">You better read this</h2>
<p class="important">Here are the things you really need to
know.</p>
<img class="important" src="scary.png" alt="This is what wi
ll happen if you don't read it.">
```

to this:

stuff.html

```
<div class="important">
  <h2>You better read this</h2>
  <p>Here are the things you really need to know.</p>
  <img src="scary.png" alt="This is what will happen if you
don't read it.">
</div>
```

Create in-line styles using `<span>` s:

different.html

```
<p>Something is <span class="special">different</span> in t
his sentence.</p>
```

Style `<div>` s and `<span>` s just like you would any other element, without explicitly using `div` or `span` in the selector:

styles.css

```
.important {
  color: red;
}
```

CSS Sizing Units

In the following video, we cover:

- `px` for pixels, which sets an element to a fixed size on the page.
- `%` for percentage, which sets an element to a percentage of its parent element.
- `vh` or `vw`, which sets an element to a percentage of the viewport's width or height. The **viewport** is the area of a

webpage that's visible to a user.

- `em` or `rem`. Using `em` sets the size of an HTML element relative to its parent element's font-size. Using `rem` sets the size of an HTML element relative to the root element's font-size. We won't use `em` or `rem` in the program, so further exploration is up to you.
- We'll also learn how `%`, `vh`, `vw`, `em`, and `rem` are sizing units that are **responsive**, which means they are not fixed sizes, but change based on the size of HTML elements that they are relative to.

03:05

Previous (/introduction-to-programming/git-html-and-css/practice-styling-with-classes-and-ids)
Next (/introduction-to-programming/git-html-and-css/practice-html-divs-and-spans-and-css-sizing-units)

Lesson 41 of 64
Last updated March 24, 2023

disable dark mode

Epicodus (http://www.epicodus.com)

Lesson Wednesday

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / Using Floats

Text Cheat sheet

Terminology

- **Float**: A CSS option that assists in layout by allowing elements on the page to be placed next to each other in the layout (values include right or left).

Examples

Use floats to make elements float to one side of the page. Other elements will wrap around that element.

Float an image to the left:

floating-image.html

```
<img class="main" src="img/walrus.jpg" alt="A walrus baskin
g on a rock.">
```

styles.css

```
img.main {
  float: left;
}
```

Create a sidebar:

sidebar.html

```
<h1>Title</h1>
<div class="sidebar">
  <h2>Sidebar</h2>
  <p>Some sidebar stuff.</p>
</div>
<p>Main page stuff.</p>
```

sidebar-styles.css

```
.sidebar {
  float: right;
}
```

Make a column layout:

columns.html

```
<div class="column">
  <p>Column 1</p>
</div>
<div class="column">
  <p>Column 2</p>
</div>
<div class="column">
  <p>Column 3</p>
</div>
```

column-styles.css

```
.column {
  width: 300px;
  float: left;
}
```

Previous (/introduction-to-programming/git-html-and-css/practice-html-divs-and-spans-and-css-sizing-units)
Next (/introduction-to-programming/git-html-and-css/practice-using-floats)

Lesson 43 of 64
Last updated March 24, 2023

disable dark mode

(http://www.epicodus.com)

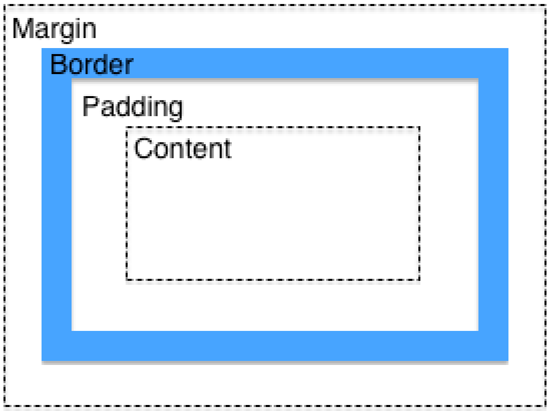
Lesson Wednesday

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / Box Model

Text Cheat sheet

Terminology

- **Box model**: A model that shows the properties of how the space around an element is assigned and manipulated with CSS.



Example

```
.my-class {
  margin: 12px;
  border: 2px solid green;
  padding: 4px;
  background-color: orange;
  box-sizing: border-box;
}
```

A Visual Review of the Box Model

The following video uses a simple HTML document with some CSS styling to practice:

- Using the browser's developer tools to inspect the box model for individual HTML elements.
- Updating the CSS for these elements to change their padding, border, and margin.

01:18

Previous (/introduction-to-programming/git-html-and-css/practice-using-floats)

Next (/introduction-to-programming/git-html-and-css/practice-box-model)

Lesson 45 of 64
Last updated March 24, 2023

disable dark mode

(http://www.epicodus.com)

Lesson Thursday

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / Cascading

Text Cheat sheet

Terminology

- **Inheritance**: The process by which children elements share the properties of their parent elements. The **cascading** in cascading style sheets also refers to this phenomenon.
- **Specificity**: In the instance that two styles conflict, the style that is applied will be the most specific of the two. For example, a style for a paragraph tag would not be applied if there was a style for a link within a paragraph. The link within the paragraph reference is more specific than paragraph.
- **Inline style**: Style that is added directly into the HTML for an element. Inline styling takes precedence over styling in your CSS files. It is not considered good practice to style HTML this way.

Examples

Elements inherit styles from their parents. This rule gives every element on the page the San Serif font:

```
body {
  font-family: sans-serif;
}
```

You can nest selectors:

nesting-styles.css

```
.sidebar p img{
  display: block;
}
```

nesting.html

```
<div class="sidebar">
  <p>This image: <img src="whatever.jpg" alt="whatever"> will be on its own line.</p>
</div>
```

More specific rules take precedence over less specific rules:

specificity-styles.css

```
p {
  background-color: green;
}

p.slow {
  background-color: yellow;
}

p.slow .stop {
  background-color: red;
}
```

specificity.html

```
<p>This will have a green background.</p>
<p class="slow">This will have a yellow background. <span class="stop">And this a red background.</span></p>
```

Tips

- In the case of conflicting rules, the last one wins. There are more complicated rules that we won't get into.

Previous (/introduction-to-programming/git-html-and-css/weekend-homework-and-preparing-for-the-next-section)
Next (/introduction-to-programming/git-html-and-css/practice-cascading)

Lesson 53 of 64
Last updated March 24, 2023

disable dark mode

(http://www.epicodus.com)

Lesson Thursday

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / CSS Media Queries and Responsive Design

Text Cheat sheet

Terminology

- **Media query**: A block of CSS applied only when certain conditions about the size or type of screen/window a user is viewing our content with are true. (i.e.: CSS associated with a media query with a max-width of 500px would only apply its styles when the viewport width is below 500px).
- **Breakpoint**: The point at which a media query's condition becomes true. A media query with a max-width of 500px will apply its styles when the viewport is less than 500px. 500px is then the break point.
- **Viewport**: The tool a user is viewing content with. Usually a browser window on a computer, phone, or tablet.
- **Responsive web design**: An approach to web design focused around providing the best viewing and navigation experience for the specific device and screen size a user is viewing content with.

- **Media types**: The type of media device the user is viewing content with (such as print, screen, handheld, or all)
- **Media features**: Specific properties about the manner the user is viewing content; including width and height of viewport, orientation of device, etc.

Examples

The following query will make the background red and text white when the browser window has a width below 768px:

```
@media screen and (max-width:  768px)  {
    body {
        background:  #FF6347;
        color: #FFF;
    }
}
```

In the code above:

- `screen` is the **media type**.
- `768px` is the **breakpoint**.
- `max-width` is a **media feature**.
- `body` is a **selector**.
- `background` and `color` are **properties**.
- `#FF6347` and `#FFF` are **values**.

Additional Resources

- Mozilla Developer Network — Using media queries (https://developer.mozilla.org/en-US/docs/Web/CSS/Media_Queries/Using_media_queries)
- Mozilla Developer Network — @media (https://developer.mozilla.org/en-US/docs/Web/CSS/@media)

- Example usages of media queries — Mediaqueri.es (http://mediaqueri.es/)

Previous (/introduction-to-programming/git-html-and-css/video-recap-core-css-concepts-and-debugging)
Next (/introduction-to-programming/git-html-and-css/practice-css-media-queries-and-responsive-design)

Lesson 56 of 64
Last updated March 24, 2023

disable dark mode

Epicodus (http://www.epicodus.com)

Lesson Thursday

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / Introduction to Bootstrap and Front-End Frameworks

Text Cheat sheet

Terminology

- **Front-End**: The portion of a website or application that is in the browser.
- **Back-End**: Code that serves data or manages a database that cannot be accessed by a user.
- **Framework:** A foundation of pre-built tools, software, code, and other resources we can use to create a project more easily than if we created it from scratch.

Library. A library is a smaller set of pre-defined code that we can download and use in our apps. It is quite similar to a framework, though frameworks are always larger and often comprise multiple libraries and other tooling.

Toolkit. A toolkit is a set of pre-built tools used for a particular purpose. Often, a toolkit is defined as multiple and related libraries, packaged into one. Usually, frameworks are much larger than toolkits, and a toolkit is larger than a library.

- **Front-End Frameworks:** Pre-built code that assists us in creating an application that runs in the browser.
- **Bootstrap**: One of the most widely-used free front-end frameworks.

Additional Resources

- Expo of Sites Using Bootstrap (https://expo.getbootstrap.com/)
- Sites Built with Bootstrap from Awwwards (https://www.awwwards.com/websites/bootstrap/)
- Bootstrap's Website (https://getbootstrap.com/)

Previous (/introduction-to-programming/git-html-and-css/practice-css-media-queries-and-responsive-design)
Next (/introduction-to-programming/git-html-and-css/downloading-and-installing-bootstrap)

Lesson 58 of 64
Last updated March 24, 2023

disable dark mode

(http://www.epicodus.com)

Lesson Thursday

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / Downloading and Installing Bootstrap

Text Cheat sheet

Terminology

- **Minified**: A file where unnecessary characters have been removed, making the file smaller and more efficient. Because whitespace is removed and words are reduced to single letters, this code is hard for humans to read. Minified files generally include a `.min` in their name.
- **CDN**: Short for "content delivery network", a network of servers that deliver content to users. We can load Bootstrap into a project by linking to its CDN location. This simply links to an online version of Bootstrap's stylesheet(s) instead of a version we download and place into our project's directory.
- **End of Life**: When a software version is no longer supported by a company. This usually means that it is not receiving security fixes or new features.
- **Breaking Changes**: When there are differences between two versions of the same software that can cause your project to stop working.

Tips

- The downside to installing Bootstrap with a CDN link is that you'll be unable to see Bootstrap styles in your project without an internet connection, because the project needs to load the online stylesheet to use Bootstrap's CSS.
- For now, we'll use either the `bootstrap.css` file, or the `bootstrap.min.css` files in our projects.
- To optionally load Bootstrap's JavaScript, use the `bootstrap.bundle.js` file.

Previous (/introduction-to-programming/git-html-and-css/introduction-to-bootstrap-and-front-end-frameworks)
Next (/introduction-to-programming/git-html-and-css/bootstrap-basic-elements)

Lesson 59 of 64

Last updated March 24, 2023

disable dark mode

Epicodus (http://www.epicodus.com)

Lesson Thursday

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / Bootstrap Basic Elements

Text Cheat sheet

Overview

Here's a brief overview of the Bootstrap elements covered in this lesson:

Container

A container (https://getbootstrap.com/docs/5.2/layout/containers/) wraps content and adds padding. We can implement a container like this:

```
<div class="container">
  ...
</div>
```

It's generally recommended to house the entirety of a document's `<body>` in a container, to provide padding to all content.

Blockquotes

blockquotes (https://getbootstrap.com/docs/5.2/content/typography/#blockquotes) are used to create nicely-formatted quote blocks. We can implement them like this:

```
<figure>
   <blockquote class="blockquote">
      <p>"It's been really good working with you!"</p>
   </blockquote>
   <figcaption class="blockquote-footer">
      My partner the first day
   </figcaption>
</figure>
```

Built-in Classes

We can use built-in classes to add formatting to almost any Bootstrap element. There are many different helper classes. One example is adding background color, which we can do like this:

```
<p class="bg-success">This text has a green background!</p>
<p class="bg-danger">This text has a red background!</p>
```

To see all of the built-in classes for background colors, visit the background color documentation (https://getbootstrap.com/docs/5.2/utilities/background/#background-color).

To see a list of the ways that we can style text, visit the text color documentation (https://getbootstrap.com/docs/5.2/utilities/colors/) and the more general documentation for text (https://getbootstrap.com/docs/5.2/utilities/text/).

Cards

card (https://getbootstrap.com/docs/5.2/components/card/) add a visual effect that makes content appear inset or embossed into the page. We can create a card like this:

```
<div class="card">
   <div class="card-body">
   <h5 class="card-title">Command Line</h5>
   <ul class="card-text">
      <li>Navigating my documents through the command line.
</li>
      <li>Creating new files and folders through the comman
d line.</li>
      <li>Moving and deleting files and folders through the
command line.</li>
      <li>Retrieving my current location from the command l
ine.</li>
   </ul>
   </div>
</div>
```

There are other things we can add to a card, likes images and links, so make sure to check out the documentation for more information.

Previous (/introduction-to-programming/git-html-and-css/downloading-and-installing-bootstrap)
Next (/introduction-to-programming/git-html-and-css/bootstrap-navigating-documentation)
Lesson 60 of 64
Last updated March 25, 2023

disable dark mode

Epicodus (http://www.epicodus.com)

Exercise Monday

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / Practice: Git Project Setup

Text Cheat sheet

Terminology

- **Initialize**: In Git, we can initialize a new, empty repository to track changes to the project directory by running `git init`. We should always run this command in the top level of the project directory.
- **Global**: A configuration option that refers to every directory in every location of the device.
- **Hidden files**: These are files on your machine that are not listed with an `ls` terminal command. Instead, you can see them with the `ls -a` terminal command. The `.git` directory is hidden by default.

Daily Git Workflow

This workflow is only for students pair programming in person at Epicodus.

Set up name and email

On Personal Computers

Git needs your name and email to be able to connect local work on our machines to a remote GitHub account. To set up your Git credentials on your personal computer, you can do this just once by setting up **global** credentials that will be applied to all projects.

```
$ git config --global user.name "Padma Patil"
$ git config --global user.email padma@email.com
```

On Shared Computers On Campus at Epicodus

On shared computers on campus at Epicodus, at the beginning of each class session **one** student in a pair will set up **global** Git credentials. The instructions on how to do this are the same as above. Note that these global Git credentials will be wiped when students shut down the computers at the end of every work day.

Later on, the same student who set up the global Git credentials will use other Git commands to save the local project on their remote GitHub account. Then, the other student(s) in the pair will copy that project, saving it to their own remote GitHub account. We'll go over this workflow in detail when we are ready for this step.

Sharing Authorship on Shared Projects

Later on we will learn how to use a **commit trailer** when creating **commits** to share authorship among multiple people.

Create a new project directory with Git repository

In terminal:

```
$ cd Desktop
$ mkdir hello-world
$ cd hello-world
$ git init
```

Git Commands

- `git init` : Initializes new local Git repository.
- `git config --global user.name ___` : Globally configures Git profile for entire device (use only on your personal machine).

Previous (/introduction-to-programming/git-html-and-css/pair-programming-and-using-discord-when-practicing-the-git-workflow)
Next (/introduction-to-programming/git-html-and-css/practice-tracking-changes-with-git)

Lesson 7 of 64
Last updated March 24, 2023

disable dark mode

Epicodus (http://www.epicodus.com)

Exercise Monday

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / Practice: Tracking Changes with Git

Text Cheat sheet

Pair Commit Workflow

Add file(s) to track in Git

In terminal:

```
$ git add hello-world.html
```

or if multiple files should be added the `.` can be used to add all files:

```
$ git add .
```

Commit

In terminal:

```
$ git commit -m "add a short, descriptive present tense com
mit message here describing the changes made"
```

Updating the Master Branch to the Main Branch

Your global configuration should be set up so that all projects default to a `main` branch, not a `master` branch. See the Git Configurations (https://www.learnhowtoprogram.com/introduction-to-programming/getting-started-with-intro-to-programming/git-configurations) lesson if you need to update your global Git configuration.

You can also update the default branch in an individual project as well. After making your first commit, run the following command:

```
$ git branch -M main
```

Git Commands

In this reference, examples in brackets `[xxx]`, should be entirely replaced by what is indicated (do not leave brackets).

Set up configurations

- `git config user.name "[name]"`: Sets a SINGLE user name for all commits in one project. The command must be run in the root directory of that project. You should always use a local configuration on Epicodus machines.
- `git config user.email "[email address]"`: Sets a SINGLE user name for all commits in one project. The command must be run in the root directory of that project. You should always use a local configuration on Epicodus machines.

- `git config --global user.name "[name]"` : Sets a SINGLE user name for all commits on a global level. Only do this on your personal machine. You will only need to run this command once.
- `git config --global user.email "[email address]"` : Sets a SINGLE email for all commits on a global level. Only do this on your personal machine. You will only need to run this command once.

Committing

- `git add .` : Adds ALL files with changes to be committed.
- `git add [file]` : Adds the named file to be committed.
- `git commit -m "[message]"` : Records all of the staged files permanently to the version history; message should describe the changes finishing the phrase "This commit will...".
- Example: `git commit -m "add AJAX functionality to the comments form"`

Reviewing Git Information

- `git log` : Lists commit history for the current branch.
- `git status` : Lists the files where changes have been made to be committed.

Previous (/introduction-to-programming/git-html-and-css/practice-git-project-setup)
Next (/introduction-to-programming/git-html-and-css/git-best-practices)

Lesson 8 of 64

Last updated March 24, 2023

disable dark mode

(http://www.epicodus.com)

Exercise **Monday**

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / Practice: GitHub Remote Repositories

Text Cheat sheet

Terminology

- **Local**: In Git terms, local means located on the device that you are using.
- **Remote**: In Git terms, this means located outside of the device you are using; for example, GitHub.
- **README**: A file that provides more details information about a GitHub repository's code.
- **Push**: Copy the code from a local Git repository to a remote repository.
- **Main**: The main copy of a Git repository on a local machine, also known as the main branch. Traditionally, this was called the "master" branch but this terminology has been updated due to its negative connotations.
- **Clone**: To make a copy of a repository from Github on a local machine.
- **Origin**: The default nickname given to the GitHub remote repository when it is cloned.

Git Remote Commands

- `$ git remote add [remote name] [remote URL]` : Sets up a new remote location with a name and location.

Example: This command adds a new remote called "al" located at "https://github.com/epicodus-lessons/hello-world":

```
$ git remote add al https://github.com/epicodus-lessons/hello-world
```

- `$ git remote -v` : Shows the names and URLs for all of the remote repositories that the project's Git repository has stored.
- `$ git push [remote name] [branch name]` : Copies the code to the remote repository from the local Git repository.
- `$ git clone [remote URL]` : Copies the code and commit history from a remote repository to a local repository.

Tips

- Be sure that you are not cloning a project inside of an existing local repository. In other words, you should *not* be inside of a project directory when you run the `git clone` command. Otherwise you'll end up with one git repository inside of another.

Previous (/introduction-to-programming/git-html-and-css/git-best-practices)
Next (/introduction-to-programming/git-html-and-css/practice-command-line-git)

Lesson 10 of 64
Last updated March 24, 2023

disable dark mode

Epicodus (http://www.epicodus.com)

Lesson Monday

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / Hypertext Markup Language (HTML)

Text Cheat sheet

Terminology

- **Hypertext**: Another word for a link.
- **Markup language**: A system for adding distinct instructions for how text should be formatted. HTML is an example.
- **Angle brackets**: The symbols `<` `>` that are used in HTML to indicate instructions for the browser.
- **Tag**: Tags are used to surround an element. Each element should have an opening and closing tag. An opening tag uses angle brackets like this `<tag>`. A closing tag also includes a backslash in the angle brackets like this: `</tag>`.
- **Element**: The code inside of tags — often but not always text.

Previous (/introduction-to-programming/git-html-and-css/vs-code-workflow-pair-programming-in-person-or-with-live-share)
Next (/introduction-to-programming/git-html-and-css/html-indentation-spacing-and-comments)

Lesson 13 of 64
Last updated March 24, 2023

disable dark mode

Epicodus (http://www.epicodus.com)

Lesson Monday

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / HTML Indentation, Spacing, and Comments

Text Cheat sheet

Summary

- In HTML, the beginning and ending tags of an element on multiple lines should always be left-aligned. For example, when you see `<html lang="en-US">`, you should be able to visually scan the page straight down to find its closing tag (the same for `<head>` and `<body>`). One way to implement this is to create opening and closing tags at the same time and then add the contents.

Previous (/introduction-to-programming/git-html-and-css/hypertext-markup-language-html)
Next (/introduction-to-programming/git-html-and-css/practice-indentation-and-spacing)

Lesson 14 of 64

Last updated March 24, 2023

disable dark mode

Epicodus (http://www.epicodus.com)

Lesson Monday

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / HTML Block Elements: <p> Tags, Headings, List Elements, and More

Text Cheat sheet

Terminology

- **Block elements**: HTML elements that create a "block" in the display by appearing on their own line.
- **Tags**: HTML annotations which indicate how the text should be formatted.
- **P tag**: An HTML tag that indicates text should be formatted in a paragraph.
- **Opening tag**: An HTML tag that appears before the text that will be formatted in an HTML document. For example, the `<p>` in `<p> This is a paragraph. </p>` is the opening tag.
- **Closing tag**: An HTML tag that appears after the text that is formatted. It matches the opening tag but begins with a `/`. For example, the `</p>` in `<p> This is a paragraph. </p>` is the closing tag.
- **End tag**: An alternative name for a closing tag.

- **Heading**: An HTML tag to indicate the text being formatted is a heading. There are 6 sizes of HTML headings `<h1>` through `<h6>`.
- **Whitespace**: All of the "empty" space that includes spaces, indentation, blank lines, etc.
- **Unordered list**: A list of items that are designated with bullet points.
- **Ordered list**: A list of items designated with numbers.
- **List item**: An item in an ordered or unordered list.

HTML block Element Tags

- **`<head>`** : Head
- **`<body>`** : Body
- **`<p>`** : Paragraph
- **`<ul>`** : Unordered list
- **`<li>`** : List item (must exist within a set of `<ul>` or `<ol>` tags)
- **`<ol>`** : Ordered list
- **`<h1>`** through **`<h6>`** : Headings
- **`<title>`** : Title

Example

my-first-webpage.html

```
<!DOCTYPE html>
<html lang="en-US">
<head>
  <title>Web page #1!</title>
</head>
<body>
  <h1>My first web page</h1>
  <h2>Written with the guidance of Epicodus</h2>

  <p>This is my first web page!</p>
  <p>Isn't it nice?</p>

  <p>Here are some things I'm going to learn about coding:
</p>
  <ul>
    <li>HTML</li>
    <li>CSS</li>
    <li>JavaScript</li>
    <li>And a lot more!</li>
  </ul>
</body>
</html>
```

Previous (/introduction-to-programming/git-html-and-css/practice-indentation-and-spacing)
Next (/introduction-to-programming/git-html-and-css/html-block-elements-part-2)

Lesson 16 of 64
Last updated March 24, 2023

disable dark mode

Epicodus (http://www.epicodus.com)