Welcome to Intro To Coding @ GRAND CIRCUS

THINGS YOU NEED TO DO BEFORE CLASS BEGINS:

1. **Put your full name in the chat for check-in**
2. **Create a CodePen.io account for our mini project!*****
3. **Please complete our Welcome Survey at** grandcircus.co/welcome

*****SUPER important.
Don't be afraid to ask us if you need help!

RESOURCES FOR THE CLASS:

Slides:
grandcircus.co/itc-remote

The Starter Project:
grandcircus.co/itc-practice-remote

Student Activity Guide:
grandcircus.co/itc-guide

Front-End vs Back-End Languages

GRAND CIRCUS

- **Front-End:** HTML, CSS, JavaScript
  - What the users can see and interact with
  - Define what goes on in your web browser (the "**client side**")
  - This will be the focus of today's class
- **Back-End:** C#, Java, Python, among others
  - What stores and uses data
  - Define what happens on a web server to process information and return web pages (the "**server side**")

General Roles of Programming Languages

GRAND CIRCUS

HTML

- Hypertext Markup Language
- Describes the structure of a webpage
- Consists of tags “marking up” the page content

CSS

- Cascading Style Sheets
- Changes how the HTML is displayed in the browser -- colors, borders, alignment, etc.

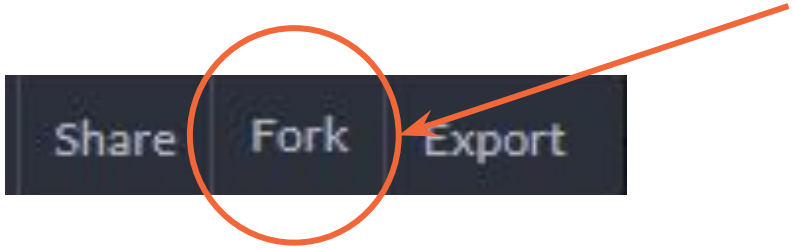
JavaScript

- Makes website content interactive
- Makes the website do things and respond to user actions

Let's Get Ready to Code!

GRAND CIRCUS

1. Go to grandcircus.co/itc-practice-remote
2. Log into CodePen, if you haven't yet
3. Click the **Fork** button (located in the bottom right corner)



There's a lot of behind-the-scenes code here --

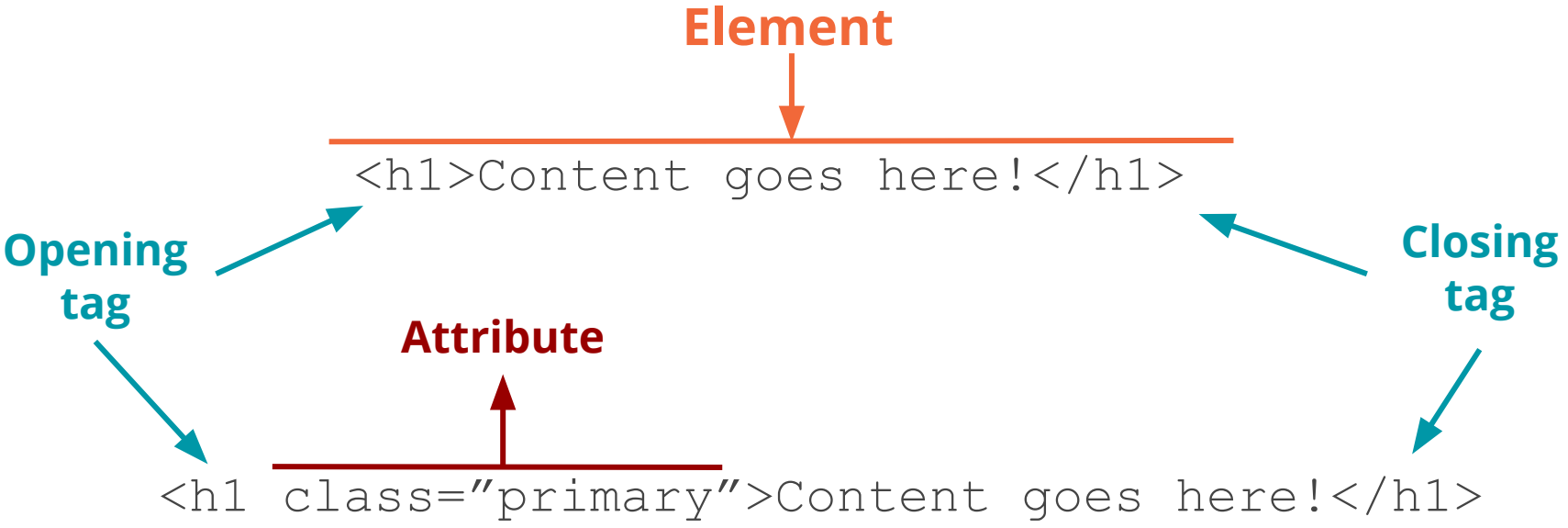
that doesn't make what we're doing "not real"

GRAND CIRCUS

HTML
(Hypertext Markup Language)

Structure and Syntax

An HTML document is made up of several units called **elements,** which are made up of **tags** and sometimes **attributes**.



While tags are essential, attributes are not needed for most tags.

Page Essentials

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <link rel="stylesheet" type="text/css" href="styles.css">

    <title>Grand Circus</title>
  </head>

  <body>
    <h1>Welcome to Grand Circus!</h1>
    <p>Happy Coding!</p>
  </body>

<html>
```

The following elements are **required** for your page to run.

<!DOCTYPE html>: Document type declaration for HTML

<html> & </html>: Everything between the opening and closing tag is a HTML document

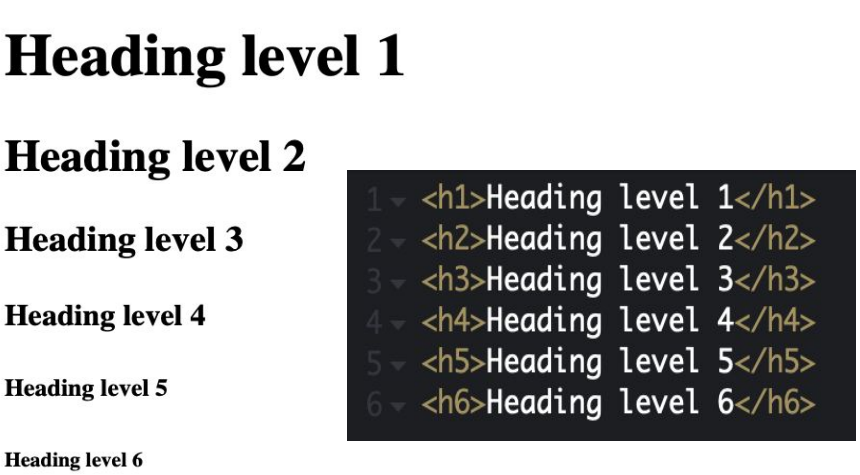
<head> & </head>: Header of the page

<body> & </body>: Visuals of the page

Basic tags

Header tags: <h1> through <h6>

- Add a header to the page
- Levels 1 through 6, size gets smaller as the number increases




```
<h1>Heading level 1</h1>
<h2>Heading level 2</h2>
<h3>Heading level 3</h3>
<h4>Heading level 4</h4>
<h5>Heading level 5</h5>
<h6>Heading level 6</h6>
```

Paragraph tags: <p>

- Allow you to section off content, similar to a paragraph in a book
- Each paragraph tag will start on a new line

```
<p>This is a paragraph.</p>
<p>This is another paragraph. Notice how this paragraph appears on a new line and sections off it own content from the first paragraph tag.</p>
```

This is a paragraph.

This is another paragraph. Notice how this paragraph appears on a new line and sections off it own content from the first paragraph tag.

Basic tags

Image tags: <img>

- Puts an image on the page
- Requires a src attribute for the file pathways
- Can use an alt attribute for alternate text
- Can also include height and weight attributes. These are defined in pixels (px), a common measurement
- Self-closing - no closing tag

```
<img src="kittens.jpg" alt="Kittens sitting in a basket" height="200" width="200">
<img src="latte.jpg" alt="a latte in a blue mug" height="240" width="247">
```

- <img src="kittens.jpg" alt="Kittens" width="200" height="200">
- <img src="smile.gif" alt="Smiling Face" width="96" height="96">
- <img src="icon.png" alt="Product Icon" width="32" height="32" >

Links/anchor tags: <a>

- Puts the “hyper” in hypertext markup language.
- Creates a link to another page or site (or even link an email or a phone number)
- Requires href attribute for URL reference
- Content in between opening and closing becomes clickable

```
<p>Hello! Click to visit <a href="http://www.grandcircus.co">Grand Circus</a> and check out our website!</p>
```

Hello! Click to visit Grand Circus and check out our website!

Basic Tags

Div tags:

- Think of it like a “division”
- Divides your content into different sections, typically for layout purposes

```
<div class="container">
  <p>This is a paragraph</p>
  <p>This is another paragraph</p>
</div>
```

Span tags:

- Holds a small piece of content, usually for styling uniquely
- Use case: highlighting one word within a paragraph or div

```
<p>This is a paragraph with a <span
class="highlight">highlighted</span> word</p>
```

This is a paragraph with a highlighted word

Basic Attributes

We covered some basic common attributes: **alt, src, href**

However, a couple more common ones you will come across are:

- **Class:** groups different HTML elements together under a common name
  - For the purpose of this workshop, the use of classes is to assign specific visual styles to multiple elements
- **Id:** help to give a unique identifier to HTML elements

These are often used for assigning specific styles (in CSS) or functionality (in JavaScript) to a group of elements or to one particular element.

```
<div id="checkout" class="hidden">
  <img id="checkout-img" height="360" width="370" align="right">
  <h2 id="selected-drink"></h2>
  <span id="price"></span>
</div>
```

```
<div id="receipt" class="hidden">
  <h2>Receipt</h2>
  ...
</div>
```

GRAND CIRCUS

CSS
(Cascading Style Sheets)

What does it do?

GRAND CIRCUS

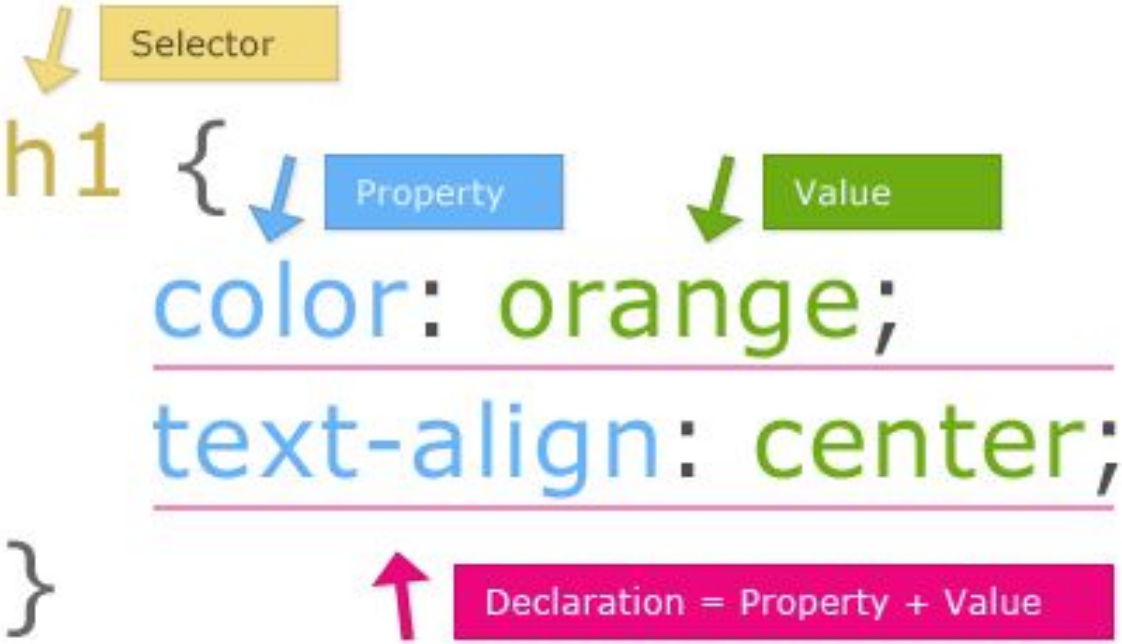
CSS controls the display of a webpage. This includes...

- Text and Background Colors
- Font and Font Size
- Other Cosmetic Aspects
- Fitting Different Screen Dimensions

CSS is the design aspect of HTML's content. Therefore, **you cannot have CSS without HTML**, as there would be no content to format.

GRAND CIRCUS

Anatomy of a CSS Rule



More About Selectors

A selector with no punctuation selects an HTML tag:

```
h1 { … }
```

A selector beginning with a period selects a class (remember that class attribute?):

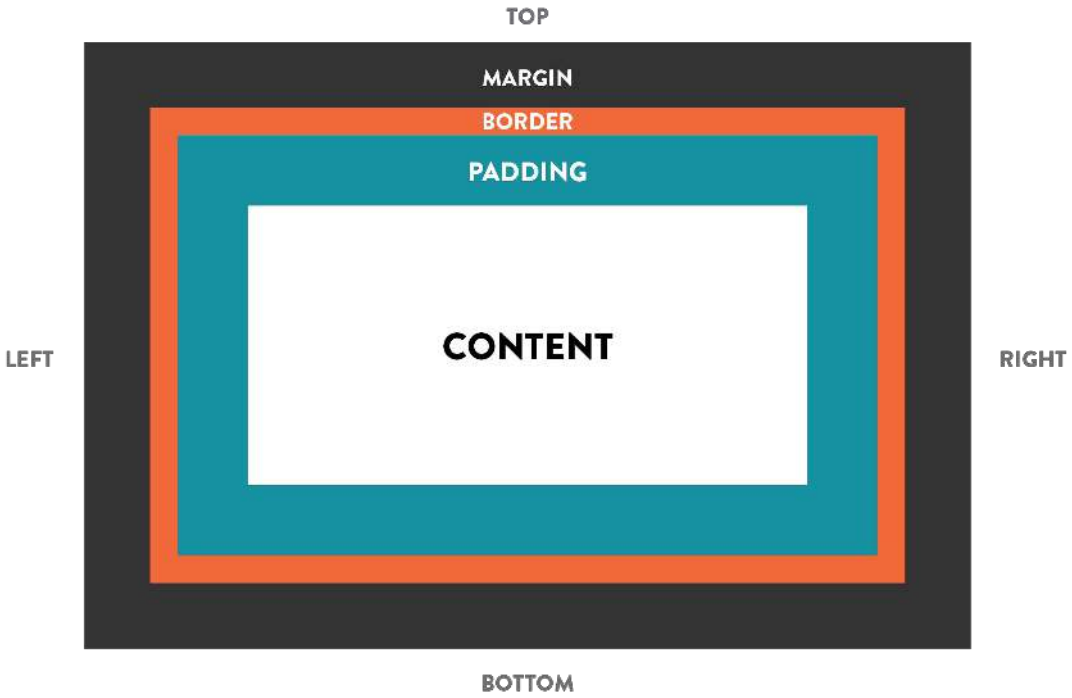
```
.hidden { … }
```

A selector beginning with a # selects an id:

```
#checkout { … }
```

CSS Box Model

Many of the properties relate to the **CSS Box Model**:



Content: What's inside the element

Border: the line (could be invisible) around the element

Margin: The space outside the border, between this element and a neighbor

Padding: The space inside the border, surrounding the content

GRAND CIRCUS

JavaScript

What does it do?

GRAND CIRCUS

JavaScript allows users to program more complex elements into a website:

- interactive graphics
- real-time updating information
- animated graphics

JavaScript puts the traditional code you may think of for apps and games into a website, making it more complex than the static code of HTML and CSS.

HTML and CSS are code. **JavaScript is programming.**

Variables

Variable: A named “container” to store data values

For example:

`x` might hold a number

`temperature` might hold “hot”, “cold”, “lukewarm”

`size` might hold “small”, “medium”, “large”

A variable’s primary purpose is to label and store data in memory for the web page to use and change, reuse at certain times, and make decisions

Creating Variables

The first time (and only the first time) we use a variable in a script, we create it with the **let** keyword.

- "let" sets up the variable and it prevents us from making some common mistakes with the variable
- This is called **declaring** a variable. You give it a name.

```
let price;          let drink;
```

We can then put a value into the variable! This is called **initializing** a variable.

- Until we do this, the variable has no value.

```
price = 4.29;          drink = "Latte"
```

Creating Variables

Although **declaring** and **initializing** are two different steps of creating a variable, you can actually do them both within the same line, like so:

```
let price = 4.29;    let drink = “Latte”;
```

Functions

GRAND CIRCUS

Functions help the computer run tasks.

- They block code together to run it at a certain time (for example, when a user clicks - this is how we will use them today)
- This helps create reusable code that we'd otherwise have to write multiple times
- They make our code simpler and make it easier to follow and read

Function Headers and Parameters

When writing a new function, we use the keyword **function**.
The function **header** names the function so we can use it later, like in our HTML

JavaScript:

```
function restart() {
  //code goes here
}
```

HTML:

```
<button onclick="restart()"
id="restart">Start Over</button>
```

A function can also receive information in the form of **parameters**.

This information can help drive the task the function is performing.

JavaScript:

```
function checkout(choice) {
  let drink = choice;
}
```

Built-In JavaScript Functions

GRAND CIRCUS

querySelector()

This function helps us reference an HTML element for use in our JavaScript code in some way.

This function will return the first element in the HTML which matches the selector passed to it.

Here, the word “document” is referencing to the HTML.

```
document.querySelector(“element name”);
```

Built-In JavaScript Functions

GRAND CIRCUS

parseInt()

As humans, we know that 1 is an **integer** (i.e, number). However, JavaScript does not explicitly know this.

Sometimes, a number might be wrapped in quotation marks (“1”), and in JavaScript’s eyes.. this is what is called a **string**, not an integer.

In order to do mathematical calculations, we need integers.

This function will attempt to turn a string of one or more characters into an integer so it can be used for calculations!

```
parseInt(“1”);      This will return you 1
parseInt(“45”);     This will return you 45
```

Conditionals

A **conditional** is a piece of code which executes if a certain expression is true.

If it's raining, I should bring an umbrella.

If it's sunny, I should bring sunglasses.

If is the keyword that tells the program to recognize a conditional

However, if a condition isn't met, you can have add other conditions as well by using **else if**.

If no conditions are true, you can have a "catch all" block of code using **else**

```
if (sunny === true){
    let item = "sunglasses"
} else if (rainy === true) {
    let item = "umbrella"
} else {
    let item = "jacket"
}
```

GRAND CIRCUS

Let us know how we did!

grandcircus.co/itc-followup

Want more?!
Check out our other workshops!

grandcircus.co/workshops

Use code ITC25 for $25 off!

Online Resources for Continued Learning!

GRAND CIRCUS

Codeacademy.com This site has tutorials in many languages. Start with HTML & CSS!

FreeCodeCamp.org

Try C#: https://www.codeschool.com/courses/try-c-sharp

Beginning Java: https://www.udemy.com/java-tutorial/