Table of Contents

Syntax Reference

What is syntax?

In coding, **syntax** is the set of rules that describe the combination and sequence of symbols (including letters and numbers) that form a correctly structured program for a specific language like JavaScript.

Symbol	Name	Example 1	Example 2
/	Forward Slash	`<body></body>`	`<img src="https://brit.co/1.jpg">`
-	Dash	`font-size: 20px;`	`$("#two").css("background-color","10px");`
" "	Quotes	`<img src = "awesome.jpg">`	`$("#div1").hide();`
' '	Single Quotes	`<img src = 'awesome.jpg'>`	`$('#div1').hide();`
< >	Angle Brackets	`<head> </head>`	`<!DOCTYPE html>`

{ }	Curly Brackets	`p {`
		`  color: blue;`
		`}`
			`function wrongAnswer() {`
			`  $("#result").show();`
			`}`

[ ]	Square Brackets	`var favColor = colors[1];`	`var colors = ["red", "blue", "yellow"];`
( )	Parentheses	`$("h1").hide();`	`wrongAnswer();`
;	Semicolon	`var word = "hello";`	`wrongAnswer();`

:	Colon	`#two {`
		`  font-size: 20px;`
		`}`
			`#two {`
			`  width: 300px;`
			`}`

.	Dot	`$(".yourclass").text("hi");`	`.yourClass {`
			`  color: red;`
			`}`

#	Hashtag	`$("#yourID").text("hi");`	`#yourID {color:red;}`

Comments	
Comments allow you to include information for other coders and is ignored by the computer.	
`<!-- These are comments in the code. -->`	Add a comment in HTML
`// One line of comments.`	Add one line comment in JavaScript
`/* Type a long section in the comments */`	Add a section of comments in JavaScript and CSS

Mathematical Operators**		
Symbol	**Definition**	**Code Example**
+	Addition****	`a + b;`
-	Subtraction	`a - b;`
*	Multiplication	`a * b;`
/	Division	`a / b;`

** Follow the order of operations rule **PEMDAS**: 1) Parentheses, 2) Exponents, 3) Multiply/Divide, 4) Add/Subtract
****Can *ALSO* be used to concatenate, or combine, strings, not just add numbers.

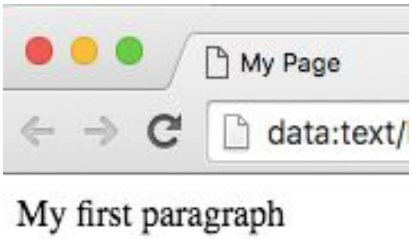
Comparison Operators		
Symbol	**Definition**	**Code Example**
<	Less than	`if (number < 10)`
>	Greater than	`else if (grade > 70)`
<=	Less than or equal to	`if (points <= 100)`
>=	Greater than or equal to	`else if (age >= 16)`
===	Equal to	`if (username === "scripted1")`
!==	NOT equal to	`else if (password !== "p@$sw0rd")`

Logical Operators		
Symbol	**Definition**	**Code Example**
&&	And	`if (number > 10 && number < 20)`
||	Or	`if (grade > 65 || passedRegents)`
!	Not	`if (!(number < 10))`

HTML

Basic Structure of an HTML document (or webpage)

```
<!DOCTYPE html>
<html>
  <head>
    <title>My Page</title>
  </head>
  <body>
    <p>My first paragraph</p>
  </body>
</html>
```



HTML Element

is an individual component of a webpage

Opening Tag ↓ `<p>` | Content ↓ `This is a paragraph` | Closing Tag ↓ `</p>`

HTML Elements	Code Example	Output
paragraph	`<p>This is a paragraph.</p>`	This is a paragraph.

heading	`<h1>Heading level 1</h1>`
	`...`
	`<h6>Heading level 6</h6>`
		Heading level 1
		Heading level 6

ordered list (w/ numbers)	`<ol>`
	`  <li>George Washington</li>`
	`  <li>John Adams</li>`
	`</ol>`
		1. George Washington
		2. John Adams

div	`<div>This is a div</div>`	This is a div
input**	`<input>`	

****Self-closing:** Does not have a closing tag.

Nesting in HTML

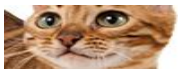
In coding, **nesting** is when you put one tag completely inside another tag's content. It allows you to organize your page's content into multiple levels.	`<div>`
	`  <h1>Weekday</h1>`
	`  <p>Monday</h1>`
	`</div>`
		On the left, the `<h1>` and `<p>` tags are nested within the `<div>` tags because the `<h1>` and `<p>` tags are completely within the opening `<div>` tag and the closing `</div>` tag.

HTML Attribute Syntax

An **attribute** adds extra information to an HTML element.
In HTML syntax, attributes are part of an HTML opening tag.

Opening Tag	Content	Closing Tag
	Google it!	

Start w/ angle bracket ↓ — attribute ↓ — End w/ angle bracket ↓

HTML elements w/ attributes	Code Example	Output
image **	`<img src="https://imgur/cats.png">`	
Link (anchor tag)	`<a href="https://www.google.com">This is a link to Google</a>`	This is a link to Google
Adding id's*	`<p id="oneID">text</p>`	text
Adding classes*	`<h1 class="aClass">text</h1>`	**text**
Input w/ placeholder**	`<input placeholder="type here">`	type here

*You can add an `id` and/or `class` to any HTML element (`<img>`, `<a>`, `<li>`, `<ul>`, etc.)
****Self-closing:** Does not have a closing tag.

id vs. class

Ids and **classes** are HTML attributes that you can add to HTML opening tags.

id	class
• Each HTML element can only have one `id`. • Each page can only have one HTML element with that `id`. • In CSS and jQuery, the symbol that you use to select an `id` is a `#` (hashtag).	• You can use the same `class` on multiple HTML elements. • You can use more than one `class` on the same HTML element. • In CSS and jQuery, the symbol that you use to select a `class` is a `.` (dot).

CSS

CSS Syntax

```
1
img {
  height: 30px;
  2   3
  border: 1px solid red;
}
```

1. **Selector:** Identifies the parts of your page that will be affected by this CSS rule. You can select using the tag name, id, or class.
2. **Property:** The thing you want to change for the element(s) you've selected. Each property should be followed by a **:** (colon) .
3. **Value:** What you want to set this property to. Each value should be followed by a **;** (semicolon) .

CSS Selectors

Selector	Symbol	Code Example	What it does
tag	none	`div { width:50px; }`	Selects every `div` and gives them a width of 50 pixels. **Other options for HTML elements** (see above in HTML elements): `p`, `body`, `h1`, `ul`, `li`, `img`, etc.
id	# hashtag	`#myID{ color:blue; }`	Selects the one HTML element with the id `myID` and changes the font color to blue.
class	. period	`.myClass{ text-align:right; }`	Selects all the HTML element(s) with class `myClass` and changes the text so it's right-aligned.

CSS Properties and Values

Change ...	Code Examples	What it does

text	`font-family: "Comic Sans";`
	`font-size: 12px;`
	`text-align: center;`
	`color: blue;`
		Changes the font to `Comic Sans`.
		Changes font size to `12` pixels.
		Aligns the text to the `center`.
		Changes the font color to `blue`.

color	`background-color: #000000;` `color: yellow;`	Changes the background color to that hex code, which is black. Changes the font color to a specific shade of `yellow`.

position	`position: relative;`
	`position: absolute;`
	`position: fixed;`
		Positions the element relative to its parent element.
		Positions the element on the page based on defined offset values.
		Positions the element in same place, even as the page is scrolled.

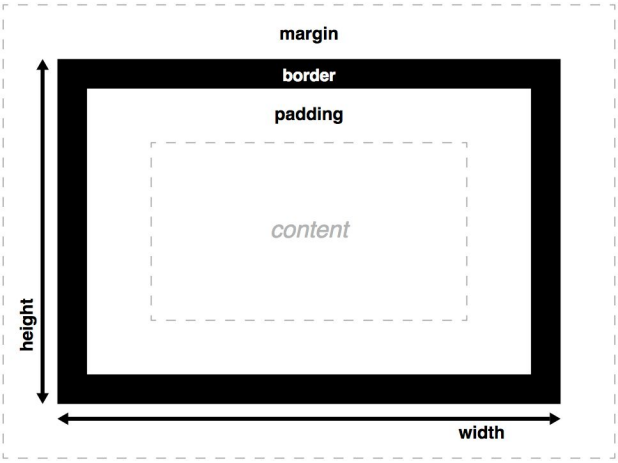
background	`background-color: pink;` `background: url("www.ex.png");`	Changes the background color to `pink`. Changes the background to an image w/ URL `"www.ex.png"`

size	`width: 50px;`
	`width: 50%;`
	`font-size: 20px;`
		Changes the width to `50` pixels.
		Changes the width to `50%` of the screen, whatever the size.
		Changes the font-size to `20` pixels.

CSS Layout

CSS Box Model

All HTML elements are shaped like boxes. Each box has a content area (text, image, link, etc.) and optional surrounding padding, border, and margin areas.



Box Model Properties

Content - (not a property) the HTML element i.e. paragraph, image, link, etc...

Padding - surrounds the content. (Example value: `10px`)

Border - surrounds the padding. (Example value: `2px black`) Think of it like an outline around a picture.

Margin - surrounds the border and buffers the content from other content. (Example value: `30px`) Margin backgrounds are always transparent.

CSS Flexbox

Flexbox is a collection of CSS properties used to align and organize HTML elements on a web page.

HTML

```
<div class="outside">
  <div class="inside">
    <h1> ScriptEd </h1>
  </div>
  <div class="inside">
    <h1> About Us </h1>
  </div>
</div>
```

Use `<div>` tags to create the structure for **flexbox** styling. The two components are:

1. The **parent element** tells the elements that are nested within how they should behave (or appear) through CSS.
2. **Child elements** should be nested inside parent elements. You must have at least one child element, but can also have more than one. They can contain any HTML element, like `<h1>` or `<p>` tags.

CSS

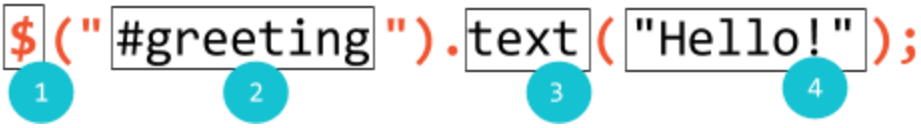
```
.outside {
    display: flex;
    flex-direction: column;
}
```

1. The **selector** should be the name of the parent element, in this case outside.
2. In the CSS, the parent element's `display` **property** MUST be set to the value `flex`. *Use this entire line of code exactly the way it is.*
3. The **flex-direction property** specifies how child elements are organized... in this example, they are organized by `column`. The other possible value is row. Other flexbox properties include `justify-content`, with possible values of `flex-start`, `flex-end`, or `center`.

jQuery

jQuery Syntax

jQuery is a JavaScript library with different actions that make it easier to make your page interactive with JavaScript.



1. The $ symbol lets you know that you are using jQuery, the JavaScript library.
2. The selector is exactly like a CSS selector. It selects or identifies the element on the page. You can use the name of an **HTML element** (`<p>`, `<h1>`, `<body>`), **id** (`#results`, `#div1`) or **class** (`.results`, `.div1`).
3. The jQuery action() to be performed on the element. See more options below.
4. The argument tells more information about how to change the element. Sometimes, there is no argument, i.e. `.show()`, and sometimes, there are several arguments, i.e. `.css()`.

Click Handler

```
1 $("#yourID").click(function(){
2   //insert code here
3   $("h1").hide();
4 });
```

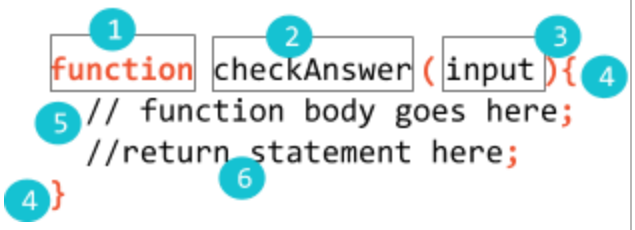
1. **When the user clicks the HTML element with an id** `yourId` **...**
2. This is a comment. The computer does not read this as code.
3. Use jQuery to hide every `<h1>` tag.
4. **End of the click handler.**

Action	Code Example	What it does
Show an element. **Hide an element.**	`$(".yourClass").show();` `$("#yourID").hide();`	Show all HTML elements w/ the class `yourClass`. Hide all HTML elements with the id `yourID`.
Replaces the content of an HTML element.	`$("body").html("<p>Hi!</p>");`	In the HTML, replace the content inside the `<body>` with `<p>Hi!</p>`.
Add/change the CSS, or style, of an element. (Change the property and/or value)	`$(".yourclass").css("color", "red");`	Add/change the CSS property `color` to `red` for all HTML elements with a class of `container`.
Add/change the text in an element.	`$("#yourID").text("You won!");`	Add/change the text to `"You won!"` for the HTML element with the id `results`.
Add/change an HTML attribute. (See page 4 for info about attributes.)	`$("img").attr("src", "http://pics.com/blah.jpg");`	Add/change the HTML attribute `src`, or source, to that URL for all `<img>` tags.
Append (add) content to an element.	`$("div").append("Bye!");`	Append, or add, the text `"Bye!"` to the end of the all the `<div>` tags.
Retrieve a value from an <input>	`var firstName = $("input").val();`	Retrieve a value from the input tag and store it in a variable named `firstName`.

JavaScript

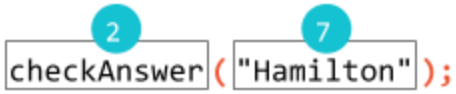
Function Syntax

A **function** is a set of instructions-- the basic building block of a program.
A **function declaration** creates the set of instructions.



1. The keyword `function` is *always* used to start a function declaration.
2. The name of this function is `checkAnswer`.
3. Some functions use parameters. The name of this parameter is `input`. You may also accept *multiple* parameters, separated by commas.
4. Curly brackets { } surround the body of the function.
5. The body of the function is the list of instructions, enclosed in the curly brackets.
6. The return statement stops the function and returns a value to the caller of the function. But, not every function has a return statement.

To use the list of instructions, you must make a **function call.**



2. To call the function, use the function name `checkAnswer`.
7. In a function call, you should pass an argument for every parameter in the function declaration. The parentheses ( ) are *always* included, even if there isn't an argument. (see above).

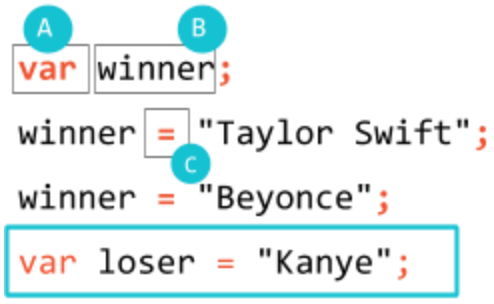
Function Example with Return Statement

```
1  function compoundWord(a,b) {
2        return a + b;
3  }
4  var word1 = compoundWord("can","not");
5
6  var word2 = compoundWord("fire","work");
```

1 Declare function `compoundWord` that takes 2 parameters.
2 Body: Return parameter `a` + parameter `b`.
3 End of function `compoundWord`.
4 Call function `compoundWord`, w/ arguments `"can"` & `"not"`.
5 Assign it to the variable `word1`. The value is `"cannot"`.
6 Call function `compoundWord`, with arguments "fire" and "work". The value of variable `word2` is `"firework`.

Variable Syntax

Variables are containers for storing data values.



Parts:
A. The keyword `var` indicates declaring a variable, or creating a new variable.
B. The variable name `winner`
C. The equal = sign assigns a value.

Line 1: Declares a variable and gives it a name `winner`.
Line 2: Assigns a value to the variable `winner`.
Line 3: Re-assigns a different value to the variable `winner`. The value of `winner` is no longer `"Taylor Swift"`. It is now `"Beyonce"`.
Line 4: A shortcut! Declares a variable named `loser` and assigns it a value `"Kanye"` all in one line of code.

Value Types		
Number	Duh... you know what a number is... No quotation marks, may start with a + or -, may include a decimal.	`var temperature = -1;` `var price = 5.99;`

String	Always inside single (' ') or double ("") quotes. Can be an empty string " ". Can include letters, spaces, symbols, numbers... as long as it's in quotes.	`var greeting = "Kevin is here!";`
		`var space = ' ';`
		`var price = "$5.99";`

Array	A list of multiple values separated by commas inside square brackets []	`var oddNumbers = [1,3,5,7,9];` `var airport = ["JFK", "LGA", "SFO"];`

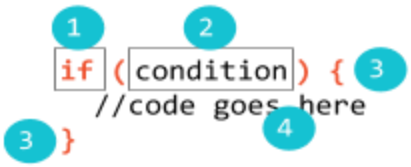
Example: Retrieve a value from an input	
`<input id="myID">` `<button id="yourID"> Go! </button>`	Creates an input field in HTML with an id `myID`. Creates a button that says Go! with an id `yourID`.

`$("#yourID").click(function(){`
`  var message = $("#myID").val();`
`});`
	When the user clicks the HTML with an id `yourID` (which is the button), retrieve the value from the input field.

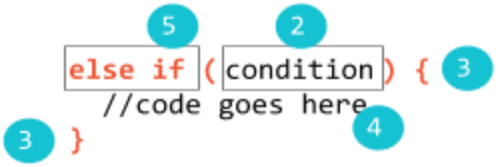
Conditional Syntax

Conditional statements are used to perform different actions based on conditions.

1) if statement



2) else-if statement



3) else statement

Conditional Statements can be created using a combination of the three statements on the left.

1. The keyword `if` indicates that this is an if statement
2. The condition goes between the ( ); the result should be true or false. If you need multiple conditions, you will need an else-if statement.
3. Curly brackets indicate the body of the condition statement.
4. Body - This is the code that executes if the condition is true. If the condition is false, then the code will NOT execute.
5. The keyword `else if` indicates an else-if statement.
6. The keyword `else` indicates an else statement.

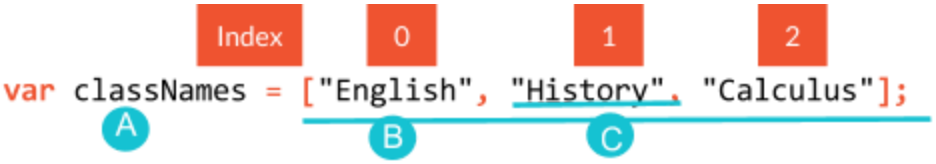
An if statement is required to create a conditional statement, while else-if statements and else statements may or may not be used. You can also use more than one else-if statement.

Basic Conditional Statement Example	

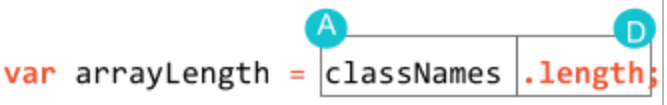
`var number = 3;`
`if (number < 5) {`
`  $("#buttonID").hide();`
`} else {`
`  $("#buttonID').show();`
`}`
	1 Declare variable named number and assign it a value of 3.
	2 If the variable number is less than 5...
	3 Hide the HTML element with the id `buttonID`.
	4 Or else...
	5 Show the HTML element with the id `buttonID`
	6 End of conditional statement.

Array Syntax

An **array** is a list-like way to store data.



A. Declare a variable called `classNames`.
B. An array is a list of values — they can be numbers, strings, or a combo. Square brackets start and end an array.
C. Each array element, or individual item (i.e. `"History"`) in the array, is separated by a comma.

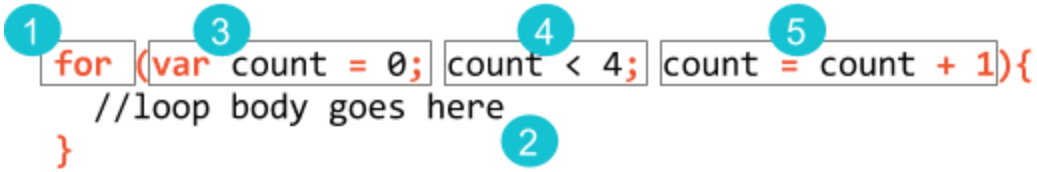


D. Arrays have properties that you can use, including length. Use the name of the array, in this case, `classNames` + `.length` to represent the length. The length of this array is 3, because there are 3 total elements in this array. The value of `arrayLength` is 3.

E. To use a specific array element, use the array index. It (see above) represents the location of an array element and always begins with 0. The array index uses the name of the array + [the index surrounded by square brackets]. The value of `favElement` is `"English"`.

For Loop Syntax

Loops repeat an action some # of times. A **for loop** repeats until a specified condition is false.



1. Always begin the for loop with the keyword `for`.
2. The loop body goes between the curly brackets. This block of code executes when the three statements are true.

The Three Parts of a 'For" Loop:

3. The 1st statement, called the Initial Expression, declares a variable and value of where the loop starts. In this case, it declares a variable `count` and begins at `0`.
4. The 2nd statement, called the Condition, tells the loop how many times to run. In this case, the loop will execute code as long as `count` is less than 4. In other words, the last time the loop will run is when `count` is 3.
5. The 3rd statement, called the Increment Expression, changes the variable value incrementally. A lot of times and in this case, the loop will increment, or increase, by 1. However, it could increment by 2 or 5 or 10, etc.

For Loop Example

```
1  for(var i=0; i=<5; i=i++){
2    $("#yourid").append(i);
3  }
```

1 Create a for loop that starts at 0, stops at 5, and increases by 1.
2 Append the value of variable `i` (0, 1, 2, 3, 4, 5) to element with id `yourid`.
3 Exit the loop when the variable `i` is no longer less than or equal to 5.

Appendix (in progress)

Shortcuts
Shortcuts are useful time-saving strategies for volunteers. They can increase your productivity, enhance your time management, and improve your professional skills and the quality of your work.

Task	PC	Mac	Explanation
Copy			
Paste			
Cut			
Undo			
Redo			

Chrome Shortcuts			

Best practices:
Indentation
Style Guide

Other Links and Resources:

- HTML
  - HTML for Beginners
- CSS
  - CSS for Beginners
  - Flexbox
  - Box Model
    - An explanation
    - How to use
  - Color/Fonts
    - Hex Color Picker - W3Schools
    - HTML Color Codes - Color Picker
    - Google Fonts
- jQuery
  - jQuery Documentation (advanced)

- JavaScript

Conditional Statement with Multiple Conditions Example

```
var num = 11;
if (num < 5) {
     console.log("Less than 5");
} else if (num < 10)  {
     console.log("Less than 10");
} else {
     console.log("Greater than 10");
}
// Greater than 10
```

Compound Conditional Statement Example

```
if (age > 16 && passedTest) {
     return "you can drive.";
} else {
     return "you can't drive.";
}
```

Complex Conditional Statement Example

```
var age = 35;
var status = "non-citizen";
if (age >= 35 && status==="citizen"){
  $("#divID").text("You can run for president");
} else if (age >= 30 && status==="citizen"){
  $("#divID").text("You can run for senate");
else if (age >= 25 && status==="citizen"){
  $("#divID").text("You can run for the House");
} else {
  $("#div").text("You can't run for office!");
}
```

1 Declare variable named age and assign it a value of 35.