ScriptEd >_

Advanced Course Reference Table

2018 Edition

Anything orange is code that always stays the same. Wherever you see black text with a gray background you can choose what to type.

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.

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();`
< >	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

Environment Set Up

Resource Linking	
`<link rel="stylesheet" href="../css/style.css">`	A style sheet is used to define the style for many HTML pages. To use an external style sheet, add a link to it in the <head> section of the HTML page.
`<script src="../script/script.js"> </script>`	A script file is used to add Javascript for many HTML pages. To use an external script, add a <script> tag at the end of the <body> tag.

Command Line	
`pwd`	The pwd command tells you in which directory you are currently located. pwd stands for print working directory.
`ls`	The ls command is used for viewing files and directories. The ls command, shows all of the major directories filed under a given file system. ls stands for list.

`cd <directory name>`
`cd ..`
`cd ~`
`cd -`
	The cd command will allow the user to change between directories. .. represents the parent directory and ~ represents the root directory. Use - to go back to the last directory. cd stands for change directory.

`mv`	The mv command - move - allows a user to move a file to another directory. Just like dragging a file located on a PC desktop to a folder stored within the "Documents". mv stands for move.
`mkdir <directory name>`	The mkdir command allows the user to make a new directory. mkdir stands for make directory.
`touch <file name>`	The touch command - a.k.a. the make file command - allows users to make files. Just as the mkdir command makes directories, the touch command makes files.
`rm` `rmdir`	The rm command like the rmdir command is meant to remove files. The rmdir command will remove directories and files within them. The rm command will delete any created files. rm stands for remove and rmdir stands for remove directory.
`clear`	The clear command clears the screen and wipes the board clean.

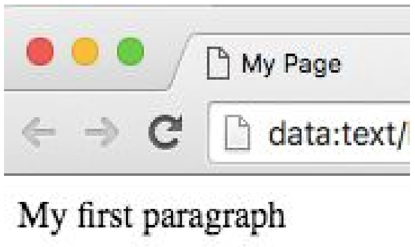
Git and Github	
Comments allow you to include information for other coders and is ignored by the computer.	
	Forking a repository creates a new copy of the repository on your GitHub profile.
`git clone <your repo link here>`	Cloning a repository to your local environment makes a local copy of your repository.

`git status`
`git add .`
`git commit -m "<your message here>"`
`git push`
	To put your local changes on GitHub, first check the status of what changes you have made, next stage all of your changes by using the add command, then commit all of your changes and finally push your changes to the repo.

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>
```



My first paragraph

HTML Element

an individual component of a webpage

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

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

heading	`<h1></h1>`
	`<h3></h3>`
	`<h6></h6>`
		`<h1>Heading level 1</h1>`
		`...`
		`<h6>Heading level 6</h6>`
			Heading level 1
			Heading level 6

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

unordered list
(with bullets)
	`<ul>`
	`    <li></li>`
	`</ul>`
		`<ul>`
		`  <li>George Washington</li>`
		`  <li>John Adams</li>`
		`</ul>`
			• George Washington
			• John Adams

button	`<button></button>`	`<button>Click Me</button>`	Click me!
div	`<div></div>`	`<div>This is a div</div>`	This is a div
input**	`<input>`	`<input>`	

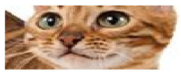
****Self-closing:** Does not have a closing tag.

Nesting and Indentation in HTML

In coding, **nesting** is when you put one tag completely inside another tag's content.
 	
Indentation helps you organize your code and makes it more readable. Remember to indent (press the tab key) when you're nesting an element inside another.
	`<div>`
	`  <h1>Weekday</h1>`
	`  <p>Monday</h1>`
	`</div>`
		On the left, the `<h1>` and `<p>` tags are nested within the `<div>` tags.

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 ↓ attribute ↓ Closing tag ↓ `<a href="www.google.com">Google it!</a>`

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

assign class in HTML	`class=" "`	`<div class="myClass">`	• The symbol that you use to select a `class` is a `.` (dot). • You can use the same `class` on multiple HTML elements. • You can use more than one `class` on the same HTML element • Classes are case-sensitive.

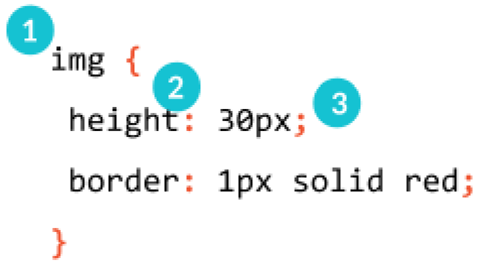
select class in CSS	`.`	`.myClass {`
		`    text-align: right;`
		`}`	

assign id in HTML	`id=" "`	`<div id="myID">`	• The symbol that you use to select an `id` is a `#` (hashtag). • Each HTML element can only have one `id`. • Each page can only have one HTML element with that `id`. • ids are case sensitive.

select id in CSS	`#`	`#myID {`
		`    color: blue;`
		`}`	

CSS

CSS Syntax



```
img {
  height: 30px;
  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) .

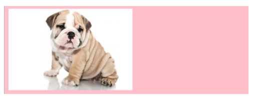
CSS Properties and Values

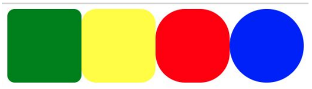
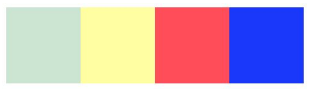
Change ...	Code Examples	Output	What it does

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

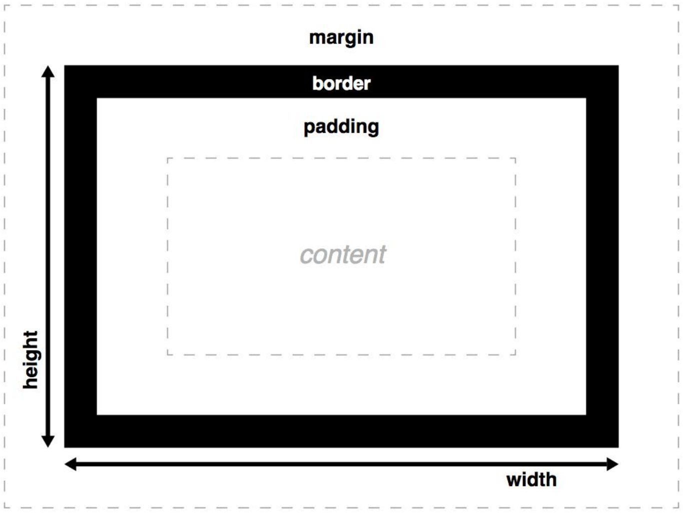
background	`background-color: pink;`
	`background: url("ex.png");`
			Changes the background color to `pink`.
			Changes the background to an image w/ URL "`www.ex.png`"

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.

border-radius	`border-radius: 500px;`		Makes the corners of a div slightly rounded
opacity	`opacity: 0.5;`		Make the whole div and all its content semi-transparent. Accepted values can be between 0 and 1.

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.

Change...	Code Examples	What it does

content	`<p>hey</p>`
	`<img src="cat.jpg">`
		Any HTML element (paragraph, image, link, etc.). *Not a property*.

padding	`padding:20px;`	Spacing between the content and border.

border	`border:20px solid red;`
	`border:10px dotted yellow;`
	`border:50px groove red;`
		Surrounds the padding. Think of it like an outline around a picture.
		Border takes 3 values that define how thick the border is, the style, and the color.

margin	`margin: 15px;`	Spacing between the border of this element and the start of another element.

If we define only one value, it will be applied to all 4 sides of the content.

`padding: 10px;`	10px padding applied to all sides

We can define a different value for all 4 sides (top, right, bottom, left).

`margin: 10px 20px 30px 40px;`	10px margin to **top** of content, 20px margin to **right** of content, 30px margin to **bottom** of content, 40px margin to **left** of content

You can define a value for a specific side of the property.

`padding-left: 100px;`
`margin-top: 25px;`
	100px padding to the left only
	25px margin to the top only

Similarly, you define a border for a specific side of the box.

`border-left: 10px solid black;`
`border-right: 20px dotted green;`
	10px solid black border to the left only
	20px dotted green border to the right only

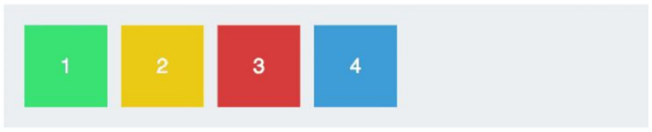
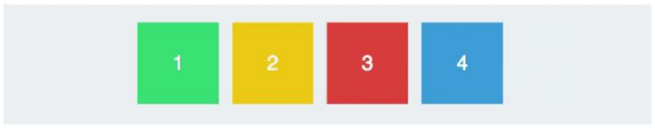
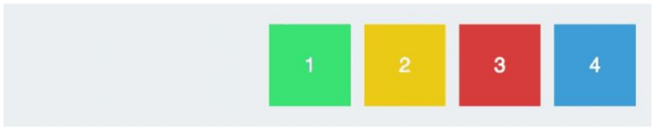
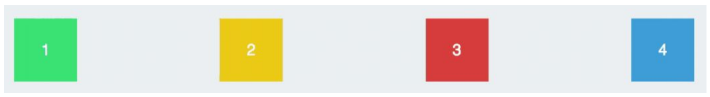
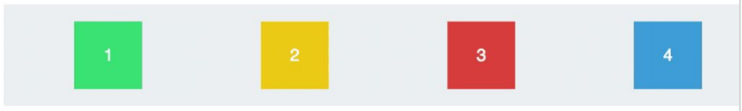
CSS Flexbox

When using flexbox, turn on **flexbox** for the parent element, using the property display and value flex.

```
.container {
  display: flex;
}
```

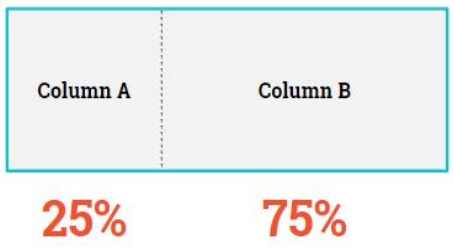
Arranged in a row

Use the **justify-content** property to align the child elements to a specific side.

Change...	Code Examples	What it does
flex-start	`.container {` `display: flex;` `justify-content: flex-start;` `}`	
center	`.container{` `display: flex;` `justify-content: center;` `}`	
flex-end	`.container {` `display: flex;` `justify-content: flex-end;` `}`	
space-between	`.container{` `display: flex;` `justify-content: space-between;` `}`	
space-around	`.container{` `display: flex;` `justify-content: space-around;` `}`	

Arranged by columns

Step 1: Turn on flexbox for the parent element (see above).
Step 2: Define the width for the child elements.

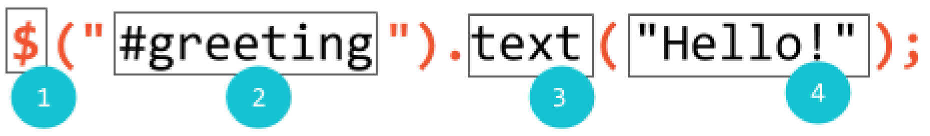


```
.section {
  display: flex;
}
.left {
  width: 25%;
}
.right {
  width: 75%;
}
```

jQuery

jQuery Syntax

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>



```
$("#greeting").text("Hello!");
```

1. The $ symbol lets the computer 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 `$(".yourClass").click(function(){`	1 When the user clicks the HTML element with a class `yourClass`
2 `$("img").hide;`	2 The HTML element `img` hides.
3 `});`	3 Closes the click handler.

Action	Code Example	What it does

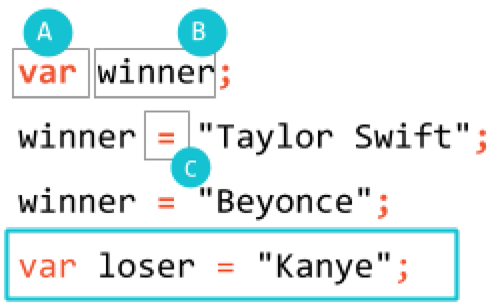
Show an element.
Hide an element.
	`$(".yourClass").show();`
	`$("#yourID").hide();`
		Show all HTML elements w/ the class `yourClass`.
		Hide all HTML elements with the id `yourID`.

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

Variable Syntax

Variables are containers for storing data values.



```
var winner;
winner = "Taylor Swift";
winner = "Beyonce";
var loser = "Kanye";
```

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 the 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";`

Boolean	true or false has no quotation marks	`var scriptedIsAmazing = true;`
		`var brunoMarsOverrated = false;`

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

Object	A collection of properties separated by commas inside curly brackets { }. A property is an association between a name (or key) and a value separated by a colon :	`var student = {`
		`name : "Erica",`
		`school : "Columbia HS",`
		`};`

Input and .val

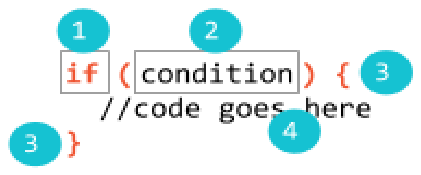
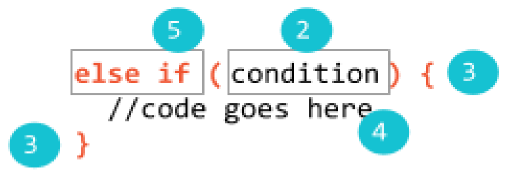
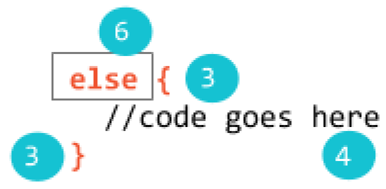
JavaScript	Code Example	What it does

input field	`<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`.

input.val	`$("#yourID").click(function(){`
	`var message = $("#myID").val();`
	`});`
		When the user clicks the HTML element 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	 `if (condition) {` `//code goes here` `}`	
2) else-if statement	 `else if (condition) {` `//code goes here` `}`	
3) else statement	 `else {` `//code goes here` `}`	

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.

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

Conditional Statement with Multiple Conditions Example

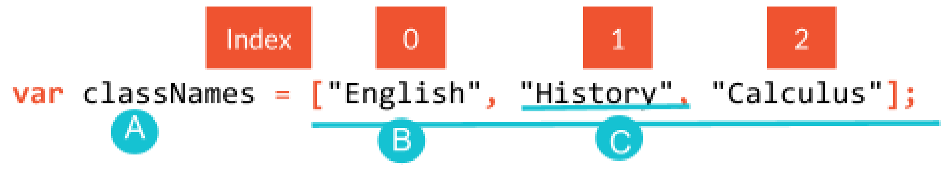
1	`var num = 11;`	1	Declare variable named num and assign it a value of 11.
2	`if (num < 5) {`	2	If the variable value is less than 5...
3	`     console.log("Less than 5");`	3	Print "Less than 5" to the console
4	`} else if (num < 10)  {`	4	Else if the number is less than 10...
5	`     console.log("Less than 10");`	5	Print "Less than 10" to the console
6	`} else {`	6	Else
7	`     console.log("Greater than 10");`	7	Print "Greater than 10 to the console
8	`}`	8	End of conditional statement.

Compound Conditional Statement Example

1	`if (age > 16 && passedTest===true) {`	1	If the value of age is greater than 16 AND passedTest is true
2	`      console.log("you can drive.");`	2	Log "you can drive." to the console.
3	`} else {`	3	Else
4	`     console.log("you can't drive.");`	4	Log "you can't drive." to the console
5	`}`	5	End of conditional statement.

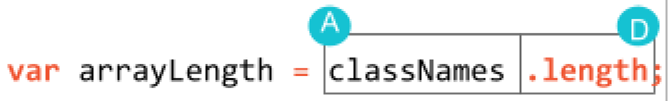
Array Syntax

An array is a way to store more than one value at a time. Think of it like a list.



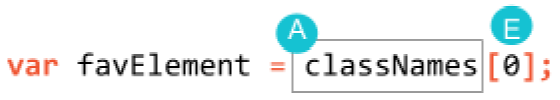
```
var classNames = ["English", "History", "Calculus"];
```

A. Declare a variable called `classNames`.
B. An array is a list of values — they can be numbers, strings,booleans or a combination of different data types. 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.



```
var arrayLength = classNames.length;
```

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.

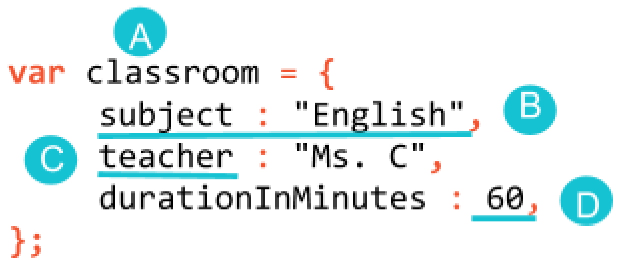


```
var favElement = classNames[0];
```

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"`.

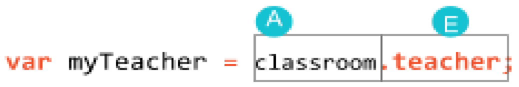
Object Syntax

An object is a way to store data as properties with keys and values.



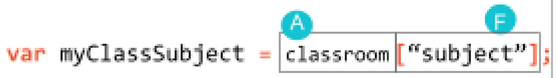
```
var classroom = {
    subject : "English",
    teacher : "Ms. C",
    durationInMinutes : 60,
};
```

A. Declare a variable called `classroom`.
B. An object is a collection of key/value pairs separated by commas.
C. Each key/value pair, in the object has a unique name or key used to identify it.
D. Each key has a corresponding value separated by a colon : .



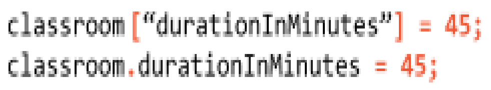
```
var myTeacher = classroom.teacher;
```

E. To access a specific value from an object in dot notation, use the corresponding key . The value of `myTeacher` is `"Ms. C"`.



```
var myClassSubject = classroom["subject"];
```

F. To access a specific value from an object in bracket notation, you can also use the corresponding key. The value of `myClassSubject` is `"English"`.



```
classroom["durationInMinutes"] = 45;
classroom.durationInMinutes = 45;
```

To change the value of a property you can use either dot notation or bracket notation and assign the property a new value.

forEach Loop Syntax

Loops repeat an action. A forEach loop repeats until all elements in an array have been selected

```
anArray.forEach(function(arrayElement) {
  //loop body goes here
});
```

1. The forEach loop is used to iterate over an array. It can be any array with any number of values or array elements
2. Iterating over an array means looping over the elements of the array and selecting each element one at a time. This variable represents the array element that is currently selected. You can name this variable anything.
3. The forEach body goes between the curly brackets. This block of code executes every time an element is selected from the array. Usually the code is doing something to the array element that is currently selected.

For Each Loop Example

```
1  var courses = ["history", "math", "science"];
2  courses.forEach(function(course) {
3    $("#schedule").append("<p>" + course + "</p>");
4  });
5
```

1 Creates an array to iterate over.
2 Uses a forEach loop to iterate over the array.
3 The variable course represents the array element that is currently selected. The first time the loop runs course is equal to "history", the second time it is "math", and the third time it is "science".

For Loop Syntax

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

```
for (var count = 0; count < 4; count = count + 1){
  //loop body goes here
}
```

1. Always begin the for loop with the keyword `for`.
2. The loop body goes between the curly brackets. This block of code executes while the condition is 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.

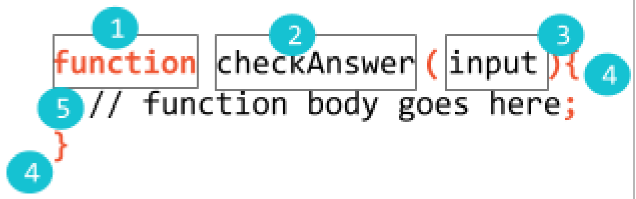
For Loop Example

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

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

Function Syntax

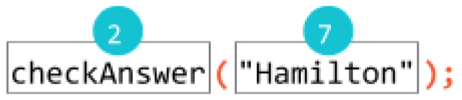
A **function** is a set of instructions-- the basic building block of a program.
A **function declaration** creates the set of instructions.



```
function checkAnswer(input){
// function body goes here;
}
```

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.

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



```
checkAnswer("Hamilton");
```

1. To call the function, use the function name `checkAnswer`.
2. 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

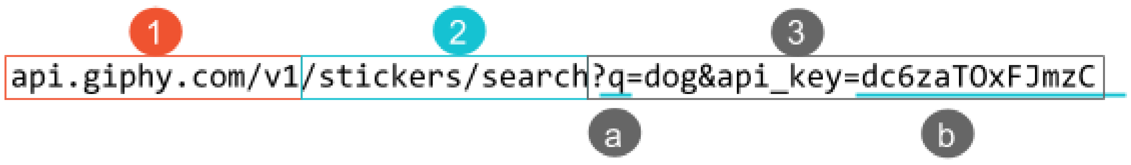
```
1  function compoundWord(a,b) {
2       return a + b;
3  }
4  var word1 = compoundWord("can","not");
5
6  var word2 = compoundWord("fire","work");
```

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"`.
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`.

APIs

API Request URL

API or Application Programming Interface Request URL

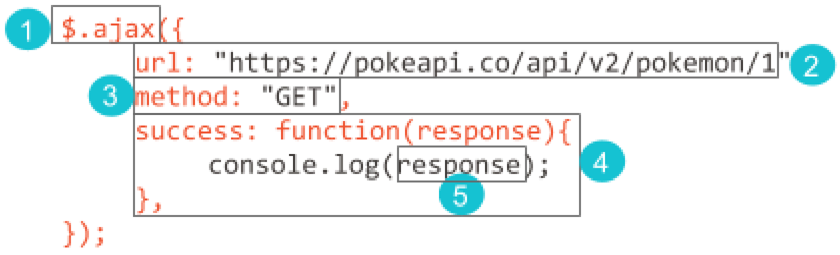


```
api.giphy.com/v1/stickers/search?q=dog&api_key=dc6zaTOxFJmzC
```

1. Base Url is the consistent part of your url. This will not change.
2. End Point refers to some object or set of objects that are exposed at an API endpoint.
3. Query String comes after the endpoint. This starts after the ? and includes the query parameters and their associated values separated by & signs.
    a. The name of the Query Parameter
    b. Value is the data that is associated with a query parameter.

AJAX Syntax

AJAX is used to retrieve data from an **API**



```
$.ajax({
    url: "https://pokeapi.co/api/v2/pokemon/1",
    method: "GET",
    success: function(response){
        console.log(response);
    },
});
```

1. Always begin the **AJAX request** with the query `$.ajax()`. The **AJAX request object** goes between the parentheses.
   Three basic properties of an AJAX request object are (there are others not listed):
2. url: Indicates where you are making the request to.
3. method: Indicates the type of request you are making. Ie. GET, POST, PUT, DELETE
4. success: the function to run upon a successful response from the API. This function takes a response as a parameter.
5. The success function uses the response object which contains all the data returned from the API call.

String Method and Properties

Action	Code Example	What it does

. length property returns the length of a string	var txt = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
	var sln = txt.length;
		Returns the length of the array. `sln` will evaluate to `26`.

.slice() extracts a part of a string and returns the extracted part in a new string.	var str = "Apple, Banana, Kiwi";
	var res = str.slice(7, 13);
		The method takes 2 parameters: the starting index (position), and the ending index (position). This example slices out a portion of a string from position 7 to position 13. The result of `res` is `“Banana”`.

A string is converted to uppercase with .toUpperCase() or to lower case with .toLowerCase():	var text1 = "Hello World!";
	var text2 = text1.toUpperCase();
	var text3 = text1.toLowerCase();
		The result of text1 is `"Hello World!"`.The value of text2 is `"HELLO WORLD!".` The value of text3 is `"hello world!"`

A string can be converted to an array with the .split() method:	var txt = "a b c d e"; txt.split(" ");	Converts `txt` from a string into an array splitting on each space. The result of `txt` is the array `["a","b","c","d","e"]`.

Array Methods and Properties

Action	Code Example	What it does

.length tells us how many items there are in the array	var fruits = ["Banana", "Orange", "Apple", "Mango"];
	var x = fruits.length;
		Returns the number of the elements in the array. `x` will evaluate to `4`.

The .pop() method removes the last element from an array:	var fruits = ["Banana", "Orange", "Apple", "Mango"];
	var x = fruits.pop();
		Removes the last element ("Mango") from fruits. The value of `x` is `["Banana", "Orange", "Apple"]`

The .push() method adds a new element to an array (at the end):	var fruits = ["Banana", "Orange", "Apple", "Mango"];
	fruits.push("Kiwi");
		Adds a new element ("Kiwi") to fruits. The result of `fruits` is `["Banana", "Orange", "Apple", “Kiwi”]`

You can re-assign an array value. Array elements are accessed using their index number:	var fruits = ["Banana", "Orange", "Apple", "Mango"];
	fruits[0] = "Kiwi";
		Changes the first element of fruits to `"Kiwi"`. The result of `fruits` is `["Kiwi", "Orange", "Apple", “Kiwi”]`

The .join() method also joins all array elements into a string.	var fruits = ["Banana", "Orange","Apple", "Mango"];
	var x = fruits.join(" * ");
		Joins all elements into a string separated by `" * "`. The result of `x` is `"Banana * Orange * Apple * Mango"`.

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

Notes