

Philly Tech Sistas

# Intro to JavaScript

Class 5

# More JavaScript!

A Quick Overview (cont.)



# Arrays

Arrays are just lists of values like words or numbers.

```
let arrayName = [element0, element1, ...];
```

You can put different types of data into an array.

```
let rainbowColors = ['Red', 'Orange', 'Blue', 'Violet'];  
let lotteryNumbers = [33, 72, 64, 18, 17, 85];  
let myFavoriteThings = ['Broccoli', 1024, 'Sherlock'];
```

Array Functions \ Properties

```
rainbowColors.length
```

## Array Functions

- Slice()
- Join()
- Push()
- indexOf()
- lastIndexOf()
- sort()

## Strings as Arrays (clarification)

- Can iterate over, use indexes BUT string doesn't inherit from Array
- Complex operations require conversion-  
Array.from(STRING), STRING.split("")

# Using Arrays

You can access array items with "bracket notation" by using the position of the object you want. Programmers start counting at zero.

```
let rainbowColors = ['Red', 'Orange', 'Yellow', 'Green', 'Blue', 'Indigo', 'Violet'];  
let firstColor = rainbowColors[0];  
let lastColor = rainbowColors[6];
```

## Changing Arrays

Use bracket notation to change an item in an array.

```
rainbowColors[0] = 'Pink';
```

## Expanding Arrays

Use "push" to add new values to an array.

```
rainbowColors.push('Purple');
```



# Exercise: Play with your food!

1. Create a food array & initialize to your favorite foods  
`let foods=['lasagne','popcorn'];`
2. Change the value of at least one position (also called index)  
`foods[1]='caramel popcorn';`
3. Add one more value to the array  
`foods.push('frosting');`
4. Print out the array & length of the array to the console  
`console.log(foods.length);`  
`console.log(foods);`

# Loops

Loops repeat tasks or allow coders to process data in lists

```
var rainbowColors = ['Red', 'Orange'];
```

```
var myWeeklyTaskList = ["Mon: See Dr", "Tue: See Mom", "Wed: See Mgr", "Th: Yell at Kids", "Fri: Spa"]
```

## For Each Loops (Arrays)

```
myWeeklyTaskList.forEach(function(task)
{
  console.log(task);
});
```

## For Loop

```
for (let i = 0; i < rainbowColors.length; i++)
{
  console.log(rainbowColors[i]);
}
```

## While Loops (Try @ Home!)

Runs the loop until the condition is met

```
let bottlesOfBeer = 99;
while (bottlesOfBeer >= 1) {
  console.log (bottlesOfBeer + ' bottles');
  bottlesOfBeer = bottlesOfBeer - 9;
}
```

## Map (Try @ Home!)

Map, applies a custom function to each array item

```
let mathArray = [1,2,3,10];
let mathResults = mathArray.map(function(val) {return
val * val});
console.log(mathResults );
```



# Exercise: Update your food

Output the food array using the for loop

```
for (let i = 0; i < foods.length; i++)  
{  
  console.log(foods[i] + " is available at index " + i);  
}
```

Output the food array using the foreach loop

```
foods.forEach(function(food) {  
  console.log(food + "<br>");  
});
```

# Object-Oriented Coding

Create your code with real world objects in mind!

## Student

First Name

Last Name

=

```
let student = {  
  firstName : "Jane",  
  lastName : "Gardener"  
};
```

```
let objectName = {  
  propertyName: propertyValue,  
  propertyName: propertyValue,  
  ...  
};
```

```
let student = {  
  firstName : "Jane",  
  lastName : "Gardener",  
  age : 20,  
  isEnrolled: true,  
  classes: ['math','science','english']  
};
```



# Accessing Objects

You can retrieve values using "dot notation"

```
let myHometown = aboutMe.hometown;
```

Or using "bracket notation" (like arrays)

```
let myHair = aboutMe['hair'];
```

Use dot or bracket notation to change properties

```
aboutMe.hair = 'blue';
```

Add new properties

```
aboutMe.pet = 'cat';
```



# Exercise: Recipes - Nailed it?!

Create an object to hold information on your favorite recipe. It should have properties for recipeTitle (a string), servings (a number), and ingredients (an array of strings).

Display your recipeTitle, servings, and ingredients list on the page.

```
let myRecipe = {  
  recipeTitle: 'Vegan pancakes',  
  servings: 4,  
  ingredients: ['rice flour', 'scallions', 'sprouts']  
}  
console.log(myRecipe.recipeTitle);  
For Home: console.log(myRecipe.ingredients);  
For Home: console.log(myRecipe.ingredients[0]);
```

# Object Methods

Objects not only hold properties/variables, but functions.

```
let myCat = {  
  furColor: 'orange',  
  sound: 'mearrrr',  
  makeNoise: function() {  
    console.log( this.sound);  
  }  
};
```

Call object methods using dot notation:

```
myCat.makeNoise();
```



# Exercise: Print Recipes

Add a function/method, printRecipe, to your recipe object

```
let myRecipe = {  
  recipeTitle: 'Vegan pancakes',  
  servings: 4,  
  ingredients: ['rice flour', 'scallions', 'sprouts'],  
  printRecipe: function() {  
    console.log(this.recipeTitle);  
    this.ingredients.forEach(function(val) {  
      console.log(val);  
    });  
  }  
}  
myRecipe.printRecipe();
```

**<BREAK />**

# Other Ways to Write to the Page <DOM>

//Writes to the web page, appending to end of the HTML\ replacing all data

```
document.write() ;
```

//Replaces content of selected id

```
<p id="p1">First Paragraph</p>
```

```
document.getElementById('p1').innerHTML = "This replaces the P text";
```

//Changes an image using the id

```
<img id="women" />
```

```
document.getElementById('women').src =
```

```
"https://live.staticflickr.com/1685/25900776992_a49895ad15_k.jpg";
```

//You can also access information Stored in the input tag

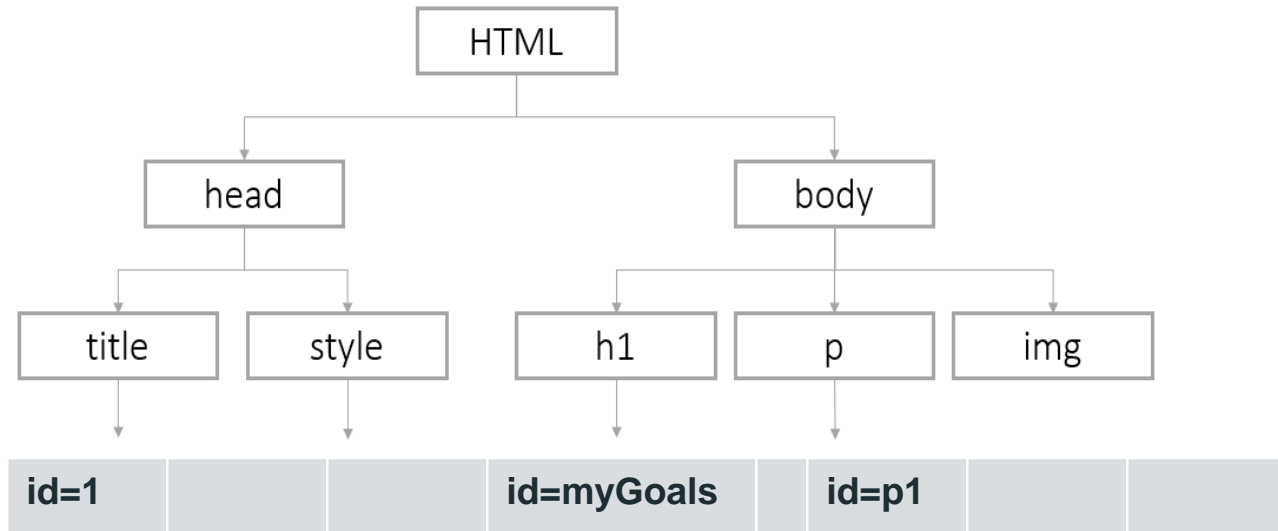
```
<input id="password" type=password value="test" />
```

```
let password = document.getElementById("password").value;
```

```
console.log(password);
```

# Accessing Elements on the Web

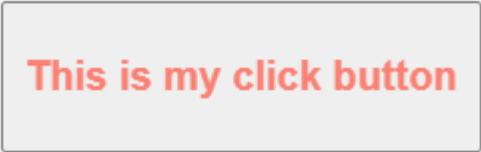
We can use the DOM to write to elements on HTML pages via their IDs



# Accessing HTML Elements via JavaScript

An '**Event**' occurs whenever a user interacts with a web page via

- mouse movements (clicks / double clicks)
- keyboard movements (down key presses and up key presses)



This is my click button

Event '**Listening**' connects a javascript function with an HTML element.

- The javascript waits until that id is used and then runs the code

```
<button id="btn">Click Me</button>
```

```
btn.onclick = function() {  
    alert("hi");  
};
```





# Exercise: Form HTML

Create an HTML form that requests user password & username

```
<form>
```

```
Username:   <input type=text id=username placeholder="##" />
```

```
Password:   <input type=password id=password placeholder="##" />
```

```
<button id=submitButton>Submit</button>
```

```
</form>
```

```
<p id=welcome>Please login</p>
```

Add a function that will handle the submit button when clicked.

```
submitButton.onclick = function(event) {
```

```
event.preventDefault(); //Stops default JS action from happening
```

```
let username = document.getElementById('username').value; //Get the username
```

```
document.getElementById('welcome').innerHTML = "Welcome " + username;
```

```
//Write out to the HTML page
```

```
}
```

# Can You handle Bro Culture? YES!

<https://genderbiasbingo.com/wp-content/uploads/2013/12/SenarioCards.pdf>

<https://www.thedoe.com/narratives/uphill-battle-against-gender-inequality-tech-industry>

# JS Objects & Modern APIs

API: Application Programming Interface

API endpoint is a url that allows you to access data about that site.

Most endpoints return data formatted as JSON, JavaScript Object Notation,

```
fetch('https://jsonplaceholder.typicode.com/posts') //calls api
.then(json => json.json() ) //gets the json
.then( (json) => {
document.write(json[1].body);
console.log(json);
})
```

# Homework





# Exercises: Events

## Create a new project

### Click Me Button

**IN HTML:** Add a button with id clickme and value of "Click Me"

```
<button id='clickme'>Click Me</button>
```

**IN JS:** Create an onclick function When user clicks the button, an alert pops up

```
clickme.onclick = function() {  
    alert("I'm Clicked");  
}
```

### Links

**IN HTML:** Add the below.

```
<a href="http://phillytechsistas.org/" id="link">Philly Tech Sistas</a>
```

When a user clicks the link, the page should display an alert instead of going to the URL. Use & modify sample code below:

```
a.onclick = function (event) {  
    event.preventDefault(); //prevents default action  
    return false; //used when event.preventDefault() doesn't work  
    let sitename = document.getElementById('link').getAttribute('href'); //gets href attribute  
    alert('If event.preventDefault had failed, we would have gone to ' + sitename);  
};
```



# Exercise: Using The DOM

Access data from HTML using CSS accessors

**IN HTML:** create h1 tag with id header and a footer tag

```
<h1 class="header">Hello</h1>  
<footer>Goodbye</footer>
```

**IN JS:** type the below

```
let header = document.querySelector('.header').innerHTML;  
let footer = document.getElementsByTagName('footer')[0].innerHTML;  
console.log( "This is the header " + header);  
console.log( "This is the footer " + footer );
```

**IN JS:** Type the below & use DOM to change the background color of the main page

```
document.getElementsByTagName('body')[0].style.backgroundColor = 'turquoise';
```



# Exercises: Loops

## Fizz Bizz

Write a program that prints the numbers from 1 to 100.

But for multiples of three print “Fizz” instead of the number and for the multiples of five print “Buzz”.

For numbers which are multiples of both three and five print “FizzBuzz”.

```
//Count from 1 to 100 for (var i = 1; i <= 100; i++) { }
```

## Odd/Even Counter

Write a for loop that will iterate from 0 to 20. For each iteration, it will check if the current number is even or odd, and report that to the screen (e.g. "2 is even")

Hint: Check out <https://philly-tech-sistas.github.io/intro-to-javascript/puzzles.html#evenodd>

## Times Table

Write a loop that gives you the 9's times table, from  $9 \times 1 = 9$  to  $9 \times 12 = 108$ .

**Bonus:** Try using a loop inside a loop to write all the times tables, from 1 to 12.



# More Coding with JS Objects

**Create a new Project**

**Declare a JS object**

```
let student = {  
  firstname: "jane",  
  lastname: "white",  
  classes: ["science", "math"],  
  gradeLevel: "Sophomore",  
}
```

**Access a property from the JS Object and print out**

```
console.log(student.gradeLevel);
```

**Use the for loop to print out ALL of the properties of the student object**

```
for (const key in student) { console.log(student[key]); }
```



# Bonus: More Recipes

Add the below code to the bottom of the JS that contains your recipe object

As a reminder, this is your recipe object

```
let myRecipe = { recipeTitle: 'Vegan pancakes', servings: 4, ingredients: ['rice flour', 'scallions', 'sprouts'] }
```

Now, add the below which will return the recipe title AND servings.

```
myRecipe.describe = function() {  
  return this.recipeTitle + ' serves ' + this.servings ;  
}  
document.write(myRecipe.describe() + "<br>");
```



# Bonus: Online Recipes

Let's make a form for users to submit recipes to our recipeBook. Add this form HTML to your page:

```
<form id="shareRecipes">
```

```
Recipe title: <input type="text" id="shareRecipeTitle" ><br>
```

```
What ingredients do you use? <br>
```

```
<input type="text" id="ingredients1" ><br>
```

```
<input type="text" id="ingredients2" ><br>
```

```
<button id="submit" type="submit">Submit</button>
```

```
</form>
```

1. Write event code that collects recipe title & ingredients when form is submitted.
2. Save the info to an object and push to the recipeBook array.
3. Use listIngredients() to print the recipe object.



# Bonus: Objects

## The Reading List

Keep track of which books you read and which books you want to read!

- Create an array of objects. Each object describes a book and has the following properties: title, author, and alreadyRead (a boolean flagging if you read the book or not).
- Iterate through the array. For each book, log to console the book title and author.
- Add an if/else statement in the body of the loop. Check alreadyRead. If true, log 'You already read "The Hobbit" by J.R.R. Tolkien', and if false, log 'You still need to read "The Lord of the Rings" by J.R.R. Tolkien.'

Hint: <https://philly-tech-sistas.github.io/intro-to-javascript/puzzles.html#reading>

**Learn More**



# Learn More

## FETCH EXAMPLES:

- <https://petehanner.medium.com/javascript-promises-and-fetch-33a5f5d13fe0>
- <https://jsfiddle.net/remotesynth/dnq2osoy/>
- <https://jsfiddle.net/omerts/75ymbnu9/>

Mozilla HTML Element Attribute: <https://developer.mozilla.org/en-US/docs/Web/HTML/Attributes>

HTML DOM Elements: [https://www.w3schools.com/jsref/dom\\_obj\\_event.asp](https://www.w3schools.com/jsref/dom_obj_event.asp)

# Free Resources to Continue Learning

Codecademy: <https://www.codecademy.com/>

- Offers a variety of courses for front end, back end, and more
- Has a free version, pro subscription, and paid 8-10 week specialized courses

Freecodecamp: <https://www.freecodecamp.org/>

- Offers curriculums for a variety of paths with certificates upon completion
- Completely free