**Object-Oriented Programming in JavaScript**

**Key Terms**

Below is a list and short description of the important keywords you have learned in this lesson. Please read through and go back and review any concepts you don't understand fully. Great Work!

|  |  |
| --- | --- |
| **Keyword** | **Description** |
| Object | A unique instance of a data structure that is defined by its class. An object comprises both data members, such as class variables and instance variables, and methods. |
| Class | A user-defined prototype for an object that defines a set of attributes that characterize an object of the class. The attributes are data members and methods, accessed via dot (.) notation. |
| Constructor | A function that is executed when the **new** operator is used to instantiate a new instance of the class. |
| Method | A special kind of function that is defined in a class body. |
| Getter | Allows access to a property that returns a value, and reflects the status of an internal variable without requiring the use of explicit method calls. |
| Setter | Can be used to execute a function whenever a specified property is attempted to be changed. Setters are most often used in conjunction with \*\*getters\*\* to create a type of pseudo-property. |
| Instance | An individual object of a certain class. |
| Inheritance | The transfer of the characteristics of a class to other class that is derived from it. |
| Extends | Creates a subclass or child of another class. |
| Super | This calls the parent function depending on the context. Most use cases will use the super() method inside of a class constructor. |

# Object-Oriented Programming in JavaScript

# Hands-On 1

Now that you have learned about JavaScript object-oriented programming, its now time to put that knowledge to work. In this Hands-On exercise, you will create a project which will solidify how these pieces operate when used together. Hands-On projects are completed outside of the browser, using your IDE (i.e., VSCode).

## Requirements

Leverage your knowledge of the JavaScript programming language to create a Stadium class. The constructor function for this class should have the following properties: a name, city, and capacity. Add a method called describeStadium() that logs out a formatted description of the stadium to the console.

* Create Stadium class.
* Add the constructor function.
* The constructor should contain three properties:
  + name - Name of the stadium
  + city - What city is the stadium located in?
  + capacity - How many fans can fit in the stadium?
* Add describeStadium() method.

If the following instance is created and the describeStadium() method is called:

let stadium1 = new Stadium("Mercedes Benz Arena", "Atlanta", 70000);

stadium1.describeStadium();

It should produce:

**OUTPUT:**

The Mercedes Benz Arena is in Atlanta and holds 70000 fans.

## Grading

* **Meets all Requirements:** 50% of your grade will be based on meeting the requirements.
* **Timely Submission:** 25% of your grade will be based on having a complete solution on time.
* **Style:** 25% of your grade will be based on having legible, and well-designed code.

Be sure to save your solution, and be prepared to share it with your Instructor or Mentor during your next class, or check-in.

# Object-Oriented Programming in JavaScript

# Hands-On 2 (optional)

Now that you have learned about JavaScript object-oriented programming, its now time to put that knowledge to work. In this Hands-On exercise, you will create a project which will solidify how these pieces operate when used together. Hands-On projects are completed outside of the browser, using your IDE (i.e., VSCode).

## Requirements

Leverage your knowledge of the JavaScript programming language to add two more methods to the Stadium class that will log which sport is mainly played in each instance of the class Stadium, and a method that changes the available seats based on the given attendance for a game.

* Create new sportPlayed() method
* Create new seatsAvailable() method

The following input given utilizing your class:

**script.js** file:

let stadium1 = new Stadium(

"Mercedes Benz Arena",

"Atlanta",

70000,

"Football",

55000

);

stadium1.describeStadium();

stadium1.sportPlayed();

stadium1.seatsAvailable();

Should produce:

The Mercedes Benz Arena is in Atlanta and holds 70000 fans.

The following sport is mainly played in this stadium: Football

There are 15000 seats still available for tonight's game.

## Grading

* **Meets all Requirements:** 50% of your grade will be based on meeting the requirements.
* **Timely Submission:** 25% of your grade will be based on having a complete solution on time.
* **Style:** 25% of your grade will be based on having legible, and well-designed code.

Be sure to save your solution, and be prepared to share it with your Instructor or Mentor during your next class, or check-in.

# Hands On 3 (optional)

Now that you have learned about JavaScript object-oriented programming, its now time to put that knowledge to work. In this Hands-On exercise, you will create a project which will solidify how these pieces operate when used together. Hands-On projects are completed outside of the browser, using your IDE (i.e., VSCode).

## Requirements

Leverage your knowledge of the JavaScript programming language to create three subclasses of the Employee parent class given. Each class should inherit from Employee as well as contain two methods specific to each subclass.

**script.js** file:

class Employee {

constructor(name, salary, hireDate) {

this.name = name;

this.salary = salary;

this.hireDate = hireDate;

}

getName() {

console.log(this.name.toUpperCase());

}

getSalary() {

console.log(this.salary);

}

getHireDate() {

console.log(this.hireDate);

}

}

Three new subclasses:

* Manager
* Designer
* QA

**Starter Code:**

class Manager extends Employee {}

Examples of the methods for these subclasses:

* jobDescription()
* yearsExperience()
* degreeCompleted()

## Grading

* **Meets all Requirements:** 50% of your grade will be based on meeting the requirements.
* **Timely Submission:** 25% of your grade will be based on having a complete solution on time.
* **Style:** 25% of your grade will be based on having legible, and well-designed code.

Be sure to save your solution, and be prepared to share it with your Instructor or Mentor during your next class, or check-in.