

Web Scripting 1

Day 06

Agenda

- Assignment 4 walkthrough
- Progress Review
- Objects
- Classes

Assignment 04

Walkthrough

Progress Check-In

- At this time you should have a rough idea of the following concepts
 - A basic understanding of:
 - Selecting Elements with JavaScript
 - DOM Manipulation
 - DOM Traversal
 - jQuery

Objects

What is an Object

- An object in computer programming can be thought of as a physical object in the real world
- Just like a physical object, a programming object has properties (such as color, size, weight) that describe its characteristics, and methods (such as turn on, turn off, open, close) that define its behavior



[This Photo](#) by Unknown Author is licensed under [CC BY-NC](#)

What is an Object

- Car Object
 - Properties
 - These are things that describe the object
 - Make
 - Model
 - Colour
 - Methods
 - These are things that an object can do
 - Start
 - Drive
 - Honk



[This Photo](#) by Unknown Author is licensed under [CC BY-NC](#)

Car Object Written with JavaScript Code

```
const car = {  
  
  // Properties  
  make: "Audi",  
  model: "Coupe",  
  year: 2020,  
  color: "red",  
  
  // Methods  
  start: function() {  
    console.log("Car started");  
  },  
  drive: function() {  
    console.log("Car is driving");  
  },  
  honk: function() {  
    console.log("Honk Honk!");  
  }  
};
```



[This Photo](#) by Unknown Author is licensed under [CC BY-NC](#)

What is an Object in Computer Programming

- In computer programming, an object is a self-contained unit of data that contains properties and methods. Some key points about objects in programming:
 - Can be instances of a class (in JavaScript these can also be constructor functions), which is a blueprint for creating objects with similar properties and methods
 - Objects have properties, which are variables that store data, and methods, which are functions that perform actions
 - Objects can interact with each other through their methods, and they can also inherit properties and methods from parent classes
 - Objects can be created and manipulated during runtime, and they can also be stored in data structures such as arrays

Built-In Objects in JavaScript

- In JavaScript, built-in objects are objects that are provided by the language itself and are available for use in any JavaScript program. These objects include things like arrays, strings, numbers, and dates, as well as more complex objects like the Math object, the Date object, and the RegExp object¹
- For JavaScript that runs in the browser, built-in objects include the Window object, the Document object, HTMLElement object

1: <https://chat.openai.com/chat>

Accessing an Object's Properties and Methods

- You can access an object's properties and methods using either dot notation or square bracket notation
- The dot notation is more common
- The square-bracket notation can be useful if you need to access a property name dynamically or the property name has characters that can not be used in JavaScript variable names such as the “-” character

Accessing an Object's Properties and Methods

Dot notation for accessing an object's properties

```
const car = { make: "Toyota", model: "Camry", year: 2020 };  
console.log(car.make); // Output: Toyota
```

Square bracket notation for accessing an object's properties

```
let car = { make: "Toyota", model: "Camry", year: 2020 };  
console.log(car["make"]); // Output: Toyota
```

Accessing an Object's Properties and Methods

Dot notation for accessing an object's methods

```
let car = {  
  make: "Toyota",  
  model: "Camry",  
  year: 2020,  
  start: function() { console.log("Car started"); }  
};  
car.start(); // Output: Car started
```

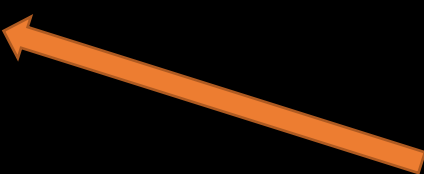


Object methods are similar to regular functions which means we call them use the method name plus "()".
The round brackets can be used for parameters

Accessing an Object's Properties and Methods

Square bracket notation for accessing an object's properties and methods

```
const stats = { points: 23, health: 78, experience: 325 }  
  
function outputScore(score, type){  
    stats[type] = stats[type] + score;  
    return stats[type];  
}  
  
outputScore(52, 'points');
```



Square brackets allow you to access an object's properties or methods dynamically by using strings that match a property's name.

When this function gets called, the variable type is equal to "points", which allows the code to access the "points" property of the stats objects.

Constructor Functions

- In JavaScript, a constructor function is a special type of function that is used to create and initialize objects
- Think of constructor functions as object factories that create objects
- A constructor function is invoked using the new keyword. When a constructor function is invoked, it creates a new object, and sets the this keyword to point to that new object

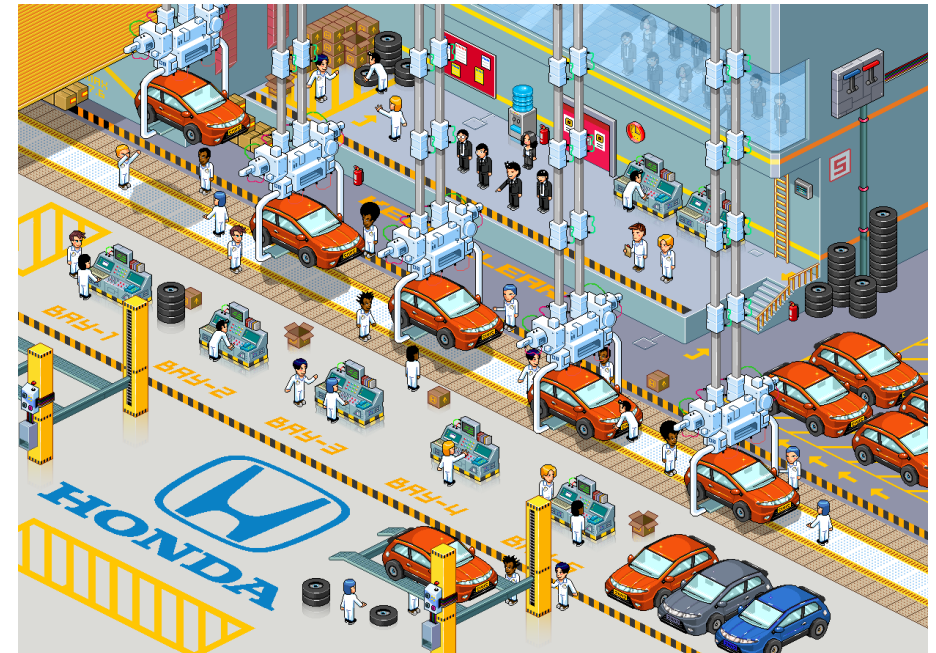
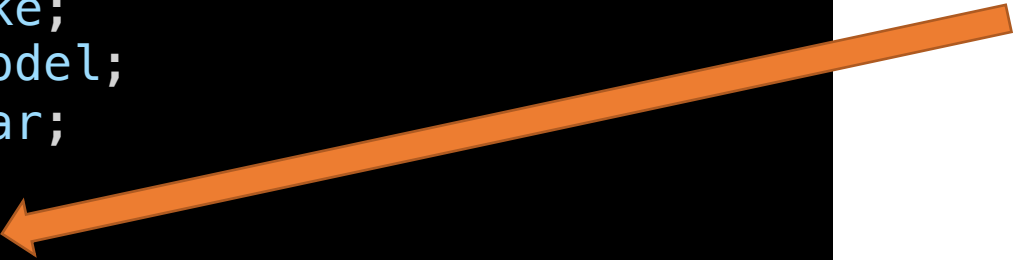


Photo from: <https://www.deviantart.com/gunstar-red/art/Honda-Factory-24239636>

Constructor Functions

```
function Car(make, model, year) {  
  this.make = make;  
  this.model = model;  
  this.year = year;  
}  
  
let myCar = new Car("Toyota", "Camry", 2020);  
  
console.log(myCar.make); // Output: Toyota  
console.log(myCar.model); // Output: Camry  
console.log(myCar.year); // Output: 2020
```



By convention, constructor functions are named with an uppercase letter at the beginning, indicating that they are constructor functions

Class Syntax

- ES6 introduced the class keyword, which is a more convenient and easier way to create constructor functions and objects
- Classes are just syntactic sugar over the prototype-based object-oriented approach of JavaScript, and the class syntax is transpiled to the constructor function and prototype-based approach
- Classes in JavaScript are not like classes in other languages like Java or C#. As mentioned above, JavaScript classes are just a shorthand for constructor functions and prototypes.

Class Syntax

```
class Car {  
    constructor(make, model, year) {  
        this.make = make;  
        this.model = model;  
        this.year = year;  
    }  
    start(){  
        console.log("Car started")  
    }  
}  
  
let myCar = new Car("Toyota", "Camry", 2020);  
myCar.start(); // Output: Car started
```

What is the Constructor Function?

- A constructor function is a special method inside a class that is automatically called when a new instance of a class is created.
- It is used to initialize the properties of the new object and perform other declared setup function
- The constructor function can accept parameters, which can be used to set the initial state of the object.
- The constructor function can only be defined once in a class.

Extending a Class

- With JavaScript class syntax you write a class that is based on a parent class
- The child class inherits all the properties and methods of the parent class
- The child class can add its own properties and methods
- This is called extending a class

Extending a Class

Parent class

```
class Animal {
  constructor(type) {
    this.type = type;
  }

  drink() {
    console.log(`The ${this.type} drinks.`);
  }

  eat() {
    console.log(`The ${this.type} eats.`);
  }
}
```

Child class

```
class Cat extends Animal {
  constructor(name) {
    super('cat');
    this.name = name;
  }

  meow(num) {
    console.log(`${this.name} meows ${num} times.`);
  }
}

const snowball = new Cat('Snowball');
snowball.eat(); // The cat eats.
snowball.drink(); // The cat drinks.
snowball.meow(3); // Snowball meows 3 times.
```

What does the super function do

- The `super()` function is used to call the parent class's constructor method
- Typically used in the constructor of a child class to inherit the properties and methods of the parent class
- The `super()` function must be called before the `this` keyword can be used in the child class