

JAVASCRIPT

Objects and Object Oriented Programming

A dark blue diagonal shape that starts from the bottom left corner and extends towards the top right corner, covering the lower right portion of the slide.

What are objects – in JavaScript

When it comes to writing code there are many ways to create a solution.

object is a collection of properties, and a property is an association between a name (key) and a value. A property's value can be a function, in which case the property is known as a method.



Object Literal notation

Objects contain data and can use that data to return results.

```
// Literal notation
var myObj1 = {
  stuff: "yes"
  , greet: "hello"
};

const person = {
  first: "Laurence"
  , last: "Svekis"
  , id: 100
  , details: function () {
    return `${this.last}, ${this.first} id#${this.id}`
  }
}
```

```
> person.details()
```

```
< "Svekis, Laurence id#100"
```

TRY IT



Create an object that reflects a car.

Add Color, make, model, price, year then for bonus points add a few methods like drive and park. Output to console.

```
> myCar
< {color: "Black", brand: "Ford", make: "Mustang", price:
  50000, year: 1965, ...}
> myCar.drive()
I'm driving Mustang, vroom vroom oopjs.html:30
```



```
const myCar = {};  
myCar.color = "Black";  
myCar.brand = "Ford";  
myCar.make = "Mustang";  
myCar.price = 50000;  
myCar.year = 1965;  
myCar.drive = function () {  
  console.log("I'm driving " + this.make + ", vroom vroom");  
}  
myCar.park = function () {  
  console.log("parking the car");  
}
```

Solution Car object

Challenge #1 Shopping CART

1. Create an array that contains a typical selection of items you might find going shopping.
2. Create JavaScript code to output it on the page as divs.
3. Add an event listener that when clicked adds the selected item to a global object and updating the quantity if the item is already there.
4. Create a method within the new object that calculates the subtotal



Milk	Apple	Bread	Butter
\$5	\$1	\$2	\$3

*You can add styling as needed to make it look nice :)

```
const items = [{
  item: "Milk"
  , id: 0
  , cost: 5
  }
  , {
  item: "Apple"
  , id: 1
  , cost: 1
  }
  , {
  item: "Bread"
  , id: 2
  , cost: 2
  }
  , {
  item: "Butter"
  , id: 3
  , cost: 3
  }
}
```

```
var cart = {};
items.forEach(function (ele) {
  let div = document.createElement('div');
  div.innerHTML = `

### ${ele.item}</h3>${ele.cost}`; div.val = ele.id; div.addEventListener('click', function () { let namer = ele.item.toLowerCase(); if (!cart[namer]) { cart[namer] = { name: ele.item , price: ele.cost , qty: 1, subtotal: function(){ return this.price * this.qty } } } else { cart[namer].qty++; } }) div.style.border = "1px solid #ddd"; div.style.display = "inline-block"; document.body.appendChild(div); })


```

Challenge #1 Shopping CART (part 2)

1. Add a cart on your page so that selected items are visible.
2. Update it as new items are added
3. Add a total to the bottom

*Go shopping and enjoy.



```
const output = document.createElement('div');
document.body.appendChild(output);
items.forEach(function (ele) {
  let div = document.createElement('div');
  div.innerHTML = `

### -${ele.item}</h3>-${ele.cost}`; div.val = ele.id; div.addEventListener('click', function () { let namer = ele.item.toLowerCase(); if (!cart[namer]) { cart[namer] = { name: ele.item , price: ele.cost , qty: 1, subtotal: function(){ return this.price * this.qty } } } else { cart[namer].qty++; } relist(); }) })


```

```
function relist() {
  output.innerHTML = "";
  console.log(cart);
  let total = 0;
  for (let pro in cart) {
    let subTotal = cart[pro].subtotal();
    total += subTotal;
    output.innerHTML += `${cart[pro].name} ${cart[pro].price} x
    ${cart[pro].qty} = ${subTotal}<br>`
    console.log(pro)
  }
  output.innerHTML += `${total}`;
}
```

JavaScript Object Oriented Programming OOP

Objects are used to model and organize code. Grouping similar items and tasks into what is known as a class.

It provide more flexibility and easier to extend on.

Class - it is the blueprint a template definition of an objects properties and methods.

JavaScript Object Constructor notation

Uses class and new to construct the object. This makes it easier to make many objects using the blueprint.

```
// Literal notation
var myObj1 = {
  stuff: "yes"
  , greet: "hello"
};
// Constructor notation
function Blueprint() {
  this.stuff = "yes";
  this.greet = "hello";
};
var myObj2 = new Blueprint();
var myObj3 = new Blueprint();
```

```
> myObj2
< ▶ Blueprint {stuff: "yes", greet: "hello"}
> myObj3
< ▶ Blueprint {stuff: "yes", greet: "hello"}
> myObj1
< ▶ {stuff: "yes", greet: "hello"}
```

```
⋮
```

TRY IT



Its back.. But now use the function to construct the object. Make a few cars..... Honda and Mustang and more if you want. It's easy :)

Also add a sell method that returns the minimum what you want to sell it based on 90% of the price. Output should match the sample.

```
> honda.sell()
```

```
I want at least $40500 for the Accord I paid  
45000
```

```
← undefined
```

```
> mustang.drive()
```

```
I'm driving Ford Mustang, vroom vroom
```

```
← undefined
```

```
> honda.park()
```

```
parking the Honda
```

```
← undefined
```

```
> mustang.sell()
```

```
I want at least $45000 for the Mustang I paid  
50000
```

```
const honda = new Car("Red", "Honda", "Accord", 45000, 2020);
const mustang = new Car("Black", "Ford", "Mustang", 50000, 1965);

function Car(color, brand, make, price, year) {
  this.color = color;
  this.brand = brand;
  this.make = make;
  this.price = price;
  this.year = year;
  this.tires = 4;
  this.drive = function () {
    console.log("I'm driving " + this.brand + " " + this.make + ", vroom vroom");
  }
  this.park = function () {
    console.log("parking the " + this.brand);
  }
  this.sell = function () {
    console.log("I want at least $" + this.price * .9 + " for the " + this.make + " I paid " + this.price);
  }
}
```

Solution Car object

Challenge #2 Dice Game

1. Create an element on the page that can be clicked
2. Create a constructor function to contain the game functions calling it DiceGame
3. DiceGame Add option to roll the dice. Math random 1-6
4. DiceGame Add option to check winner
5. In the click event add the roll for player and computer using DiceGame
6. In the click event use DiceGame object to determine winner and get result string.
7. Output the result to the clickable element

*You can add styling as needed to make it look nice :)

Player 6 vs Computer 1
Player Wins



```
const game = new DiceGame();
const dice = document.createElement('div');
dice.textContent = "Roll Here";
document.body.appendChild(dice);
dice.addEventListener('click', function () {
  let playerRoll = game.roll();
  let compRoll = game.roll();
  let winner = game.checker(playerRoll, compRoll);
  console.log(winner);
  dice.innerHTML = `Player ${playerRoll} vs Computer  
${compRoll} <br> ${winner}`;
})
```

```
function DiceGame() {
  this.roll = function () {
    this.result = Math.floor(Math.random() * 6) + 1;
    return this.result;
  }
  this.checker = function (roll1, roll2) {
    if (roll1 > roll2) {
      return 'Player Wins';
    }
    else if (roll2 > roll1) {
      return 'Computer Wins';
    }
    else {
      return 'Tie';
    }
  }
}
```

Challenge #3 Shopping Cart

- Create DOM elements for input and adding items to the store
- Add event listeners
- Create Objects for items
- Add shipping and tax to object
- Create Cart object
- Create adder method
- Create total cost method
- Create output of cart items
- Setup default item for testing

Item name Cost

new Item \$15 x 1 = \$15

Tester \$7 x 1 = \$7

Milk \$5 x 5 = \$25

Final Total \$47

Milk	new Item	Tester
\$5	\$15	\$7

*Your application should be able to add items, click new items and add them to a cart.

```
const output = document.createElement('div');
document.body.appendChild(output);
const output1 = document.createElement('div');
document.body.appendChild(output1);
const itemInput1 = document.createElement('input');
itemInput1.setAttribute('type', 'text');
itemInput1.setAttribute('placeholder', 'Item name');
output.appendChild(itemInput1);
const itemInput2 = document.createElement('input');
itemInput2.setAttribute('type', 'number');
itemInput2.setAttribute('placeholder', 'Cost');
output.appendChild(itemInput2);
const itemButton = document.createElement('button');
itemButton.textContent = "Add Item";
itemButton.addEventListener('click', addItem);
output.appendChild(itemButton);
const outputButton = document.createElement('button');
outputButton.textContent = "Output Cart";
outputButton.addEventListener('click', function () {
  cart.output();
  console.log(cart);
});
output.appendChild(outputButton);
const items = [];
const cart = new Cart();
```

```
window.onload = function () {
  itemInput1.value = "Milk";
  itemInput2.value = 5;
  addItem();
}

function addItem() {
  let tempName = itemInput1.value || "Test";
  let tempCost = itemInput2.value || 10;
  let div = document.createElement('div');
  div.innerHTML = `<h3>${tempName}</h3>${tempCost}`;
  div.style.border = "1px solid #ddd";
  div.style.display = "inline-block";
  div.style.width = "100px";
  document.body.appendChild(div);
  div.addEventListener('click', function () {
    cart.adder(tempName, tempCost);
    cart.output();
  });
  itemInput1.value = "";
  itemInput2.value = "";
}
```

Challenge #3 Shopping Cart

```

function Item(name, cost) {
  this.name = name;
  this.cost = cost;
  this.withTax = function () {
    return (this.cost * 1.15).toFixed(2);
  }
  this.shipping = function () {
    return (this.cost * 1.05).toFixed(2);
  }
}

```

```

function Cart() {
  const myList = {};
  this.totalCost = function () {
    let total = 0;
    for (let pro in myList) {
      let subTotal = myList[pro].subtotal();
      total += subTotal;
    }
    return total;
  }
}

```

```

this.output = function () {
  let total = 0;
  output1.innerHTML = "";
  for (let pro in myList) {
    let subTotal = myList[pro].subtotal();
    total += subTotal;
    output1.innerHTML += `${myList[pro].name}
    $$${myList[pro].price} x ${myList[pro].qty} = $$${subTotal}<br>`
    console.log(pro)
  }
  output1.innerHTML += `Final Total $$${total}`;
}
this.adder = function (item, cost) {
  let namer = item.toLowerCase();
  if (!myList[namer]) {
    myList[namer] = {
      name: item
      , price: cost
      , qty: 1
      , subtotal: function () {
        return this.price * this.qty } } }
  } else {
    myList[namer].qty++;
  } } }

```

Challenge #3 Shopping Cart

Congratulations on completing the section

This ebook uses <https://developer.mozilla.org/en-US/docs/Web/JavaScript> as a source for examples. Check out more about JavaScript at MDN.

Find out more about my courses at <http://www.discoveryvip.com/>

**Course instructor : Laurence Svekis -
providing online training to over
500,000 students across hundreds of
courses and many platforms.**

