ACIT 1620 - FUNDAMENTAL WEB TECHNOLOGIES

WEEK 11

NEDA CHANGIZI

Today’s Learning Outcomes

- Breakdown your program into smaller tasks called functions.
- Declare functions in JavaScript
- Call/Invoke functions in JavaScript
- Use JavaScript DOM API to access and update HTML elements and their styles.

What is a Function?

- A set of statements that performs a task or calculate a value is called a function
  - A mini-program
  - One function – one action

- What's the benefit of having a function?

- How do we use one?
  - Define it
    - Function Declaration
    - Function Expression
  - Call it

Define – Function Declaration

Name your functions something meaningful

```
function name(parameters){
   statements
}
```

JS statements

If more than one, separated by commas.

- Define and invoke a JS function called greet() that takes a string and write a welcome message to console using that string (e.g. Hello Neda!).

Define – Function Expression and Arrow Function

- A function expression can be stored in a variable.

```
let name = function(parameters){
  statements
}
```

  - Function name can be omitted in function expression -> Anonymous function.
- Re-write the function you just wrote as an anonymous function.

- Arrow functions (introduced in ES6) are a shorter way of writing function expression

```
let name = (parameters) => {
  statements
}
```

- Re-write the function you just wrote as an arrow function.

Function Parameters

```
const greet = function (student) {
    console.log(`Hello ${student}`);
};
```

- What happens if we call the function with no parameter?

```
greet();
```

- The function is expecting a parameter:

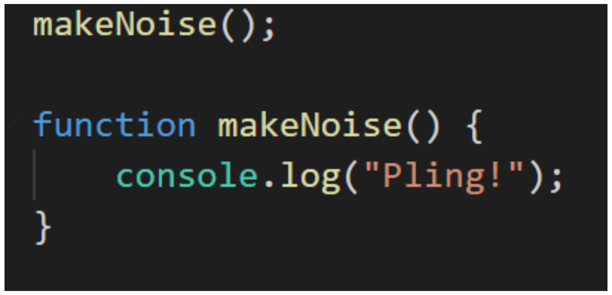
```
greet("Dan");
```

```
"Hello Dan!"
```

- The **argument** value "Dan" will be passed to our function and sits in **parameter** *student*

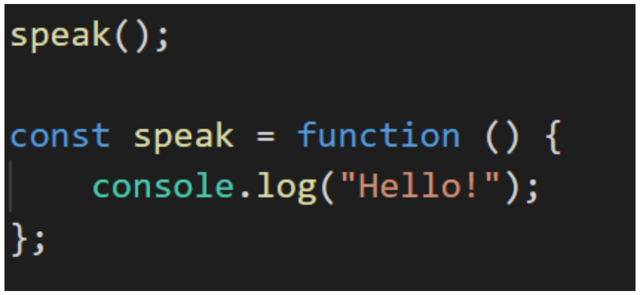
Hoisting – Function Declaration vs Expression

- Which one works?



```
makeNoise();

function makeNoise() {
    console.log("Pling!");
}
```



```
speak();

const speak = function () {
    console.log("Hello!");
};
```

✅ ❌

Function parameters – More Than One

- Update the previous function to accept another parameter ”time” which is default to ”day”. For example the greeting message could be “Good day Neda!”

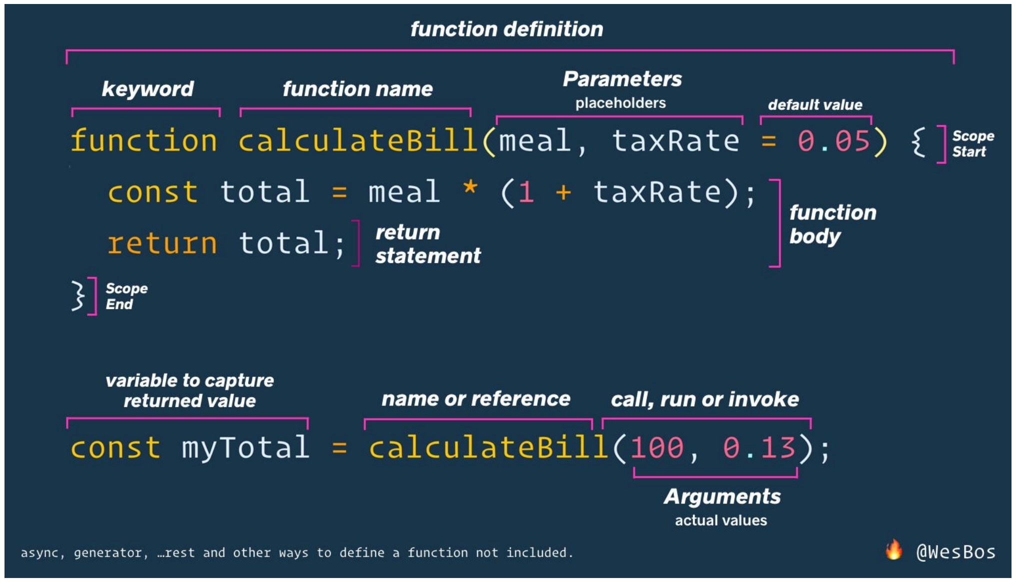
- What happens if you call the function with only one value?
  - The order matters!
- What happens if you call the function with too many variables? (more than declared)

`return` statement

- Prompt user to enter a value. Write a function that takes one parameter as radius of a circle and return the circles area. Call the area calculating function with the value entered by user. Show an alert message with both values e.g. "The area of a circle with radius 2 is 12.56"

- What happens if your function uses `return` keyword without a value?
  - By default, functions return undefined.

Function – Summary

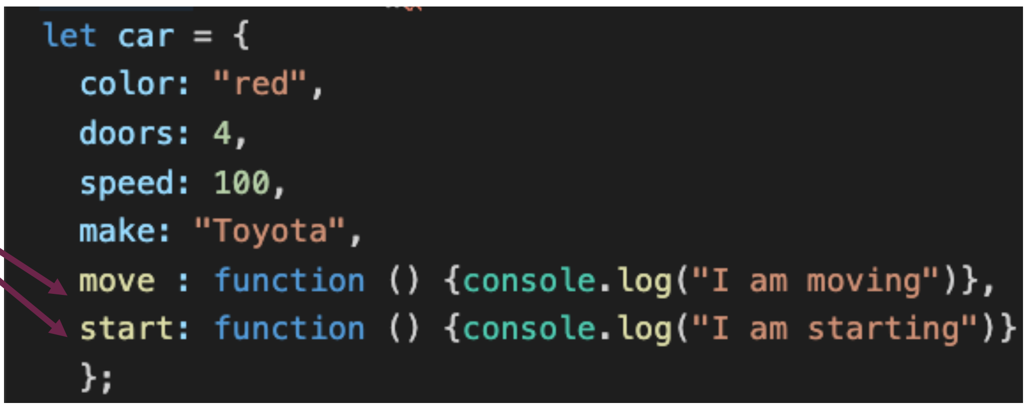


```
function calculateBill(meal, taxRate = 0.05) {
  const total = meal * (1 + taxRate);
  return total;
}

const myTotal = calculateBill(100, 0.13);
```

JavaScript Object – Review

- Properties containing a function definition are object's methods.
- To invoke a method on the car object:
  - car.move()



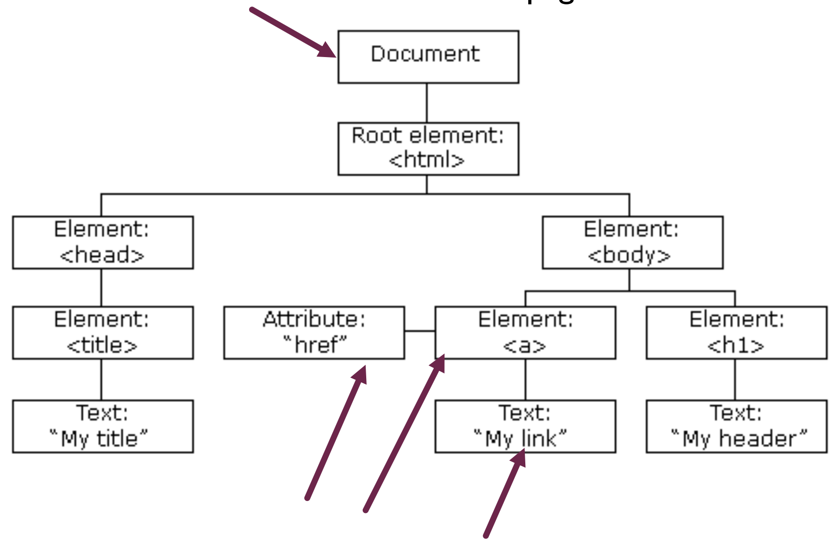
```
let car = {
  color: "red",
  doors: 4,
  speed: 100,
  make: "Toyota",
  move : function () {console.log("I am moving")},
  start: function () {console.log("I am starting")}
  };
```

Document Object Model (DOM)

- The browser reads the HTML document from the top and creates a Document Object Model of the page.
  - Tree-like structure.

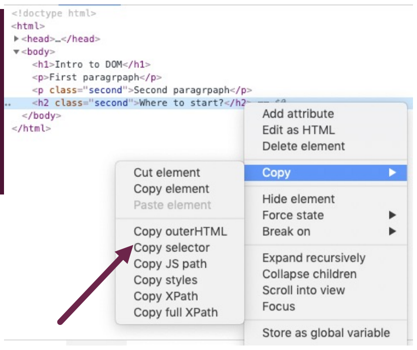
- document object has different properties and methods.

document object:
Models the HTML web page



Different kinds of nodes

Different methods to locate HTML elements



```
<!doctype html>
<html>
▶<head>…</head>
▼<body>
    <h1>Intro to DOM</h1>
    <p>First paragrpaph</p>
    <p class="second">Second paragrpaph</p>
    <h2 class="second">Where to start?</h2>
  </body>
</html>
```

- document.getElementbyId()
- document.querySelector()
- document.querySelectorAll()
- document.getElementsByClassName()
- document.getElementsByTagName()

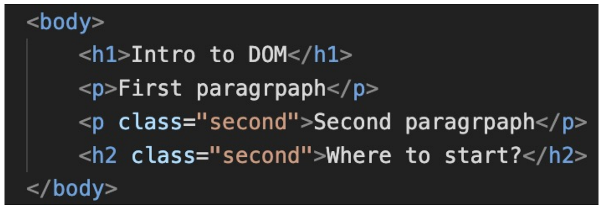
Return single element

Return NodeList

Return HTMLCollection

Similar to array.
Use [] to access items

- Example:
  - How do we access the first paragraph?
  - How about the second paragraph?
  - How about the <h2>?



```
<body>
    <h1>Intro to DOM</h1>
    <p>First paragrpaph</p>
    <p class="second">Second paragrpaph</p>
    <h2 class="second">Where to start?</h2>
</body>
```

document.getElementbyId() – textContent

- Update the area calculation function to access DOM elements and update their text according to their id:

```
<body>
   <h2>Let's try some JavaScript</h2>
   <p id="radius">Value provided by the user</p>
   <p id="result">Show area or Error</p>
</body>
```

- Update your code to use document.querySelector().
- innerText vs innerHTML vs textContent
  - Code Pen Demo

Activity - Add DOM elements

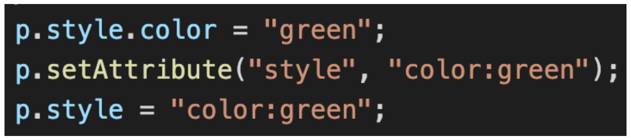
- Write a function that gets an array of strings and populate the unordered list with class="shopping" with the array elements.
  - appendChild() -> adds a node to the end of the list of children of a specified parent node.
  - You can hard-code the array in the code.
  - Do not change the html file.

```
<body>
   <ul class="shopping">Remember to buy:</ul>
</body>
```

  - Note: If the node to be added/inserted is a reference to an existing node in the document, the node will move from its current position to the new position -> A node can't be in two points of the document simultaneously

Manipulating styles - CodePen demo

- HTMLElement.style -> returns the *inline* style of an element as an object
  - **Note:** Updating the style property will completely overwrite all inline styles on an element
    - To add specific styles to an element without altering other style values, set individual properties
  - To get the values of all CSS properties for an element you should use Window.getComputedStyle() instead.
- Element.setAttribute()
  - **Note**: If the attribute already exists, the value is updated; otherwise, a new attribute is added with the specified name and value.



```
p.style.color = "green";
p.setAttribute("style", "color:green");
p.style = "color:green";
```

These two lines will remove any other **inline** style you might have

- Changing Element.classList by add(), remove(), replace(), and toggle()

Activity – Access and Change classes

- Add these rules to a css file and link it to your HTML code:

```css
.squareList {
    list-style-type: square;
}
.circleList {
    list-style-type: circle;
}
```

- Update the html to add a “circleList” class to the <ul>:

```html
<body>
    <ul class="shopping circleList">Remember to buy:</ul>
</body>
```

- Write a JS function to change the list marker type to square by using above rules and classes.

DOM – Updating attributes

- Update your HTML code:

```
<body>
   <img id="shoppingCart">
   <ul class="shopping">Remember to buy:</ul>
</body>
```

- Write a JS function to find the img tag with id="shoppingCart" and update its src, alt, width, and height attribute. You could use this link as the src: https://image.flaticon.com/icons/png/512/126/126083.png

DOM – Updating styles

- Write a JS function to find all <li> elements, check if their text contains word “green”. If so, change their text color to green.