<Semester IV>

</Front-End Development for Designers/>

Week 8

JavaScript Arrays

JavaScript Array Methods

In JavaScript, **arrays have a number of built-in methods that allow you to manipulate and operate on the elements of an array.** These methods can be used to add, remove, and modify array elements, as well as to perform common operations such as sorting and filtering.

Basic methods: length, indexOf(), push(), pop(), unshift(), shift(), includes(), slice(), splice(), concat(), includes(), split(), join(), toString(), fill(), Array.isArray(), flat()

Iteration methods: forEach(), every(), filter(), find() , map(), some(), reduce(), findIndex()

Iteration methods used to traverse and manipulate collections such as arrays, sets, and maps. Each iteration method has a distinct purpose, offering various functionalities to suit different use cases.

JAVASCRIPT ARRAY METHODS

- ARRAY.MAP()
- ARRAY.FILTER()
- ARRAY.REDUCE()
- ARRAY.REDUCERIGHT()
- ARRAY.FILL()
- ARRAY.FIND()
- ARRAY.INDEXOF()
- ARRAY.LASTINDEXOF()
- ARRAY.FINDINDEX()
- ARRAY.INCLUDES()
- ARRAY.POP()
- ARRAY.PUSH()
- ARRAY.SHIFT()
- ARRAY.UNSHIFT()
- ARRAY.SPLICE()
- ARRAY.SLICE()
- ARRAY.JOIN()
- ARRAY.REVERSE()
- ARRAY.SORT()
- ARRAY.SOME()
- ARRAY.EVERY()
- ARRAY.FROM()
- ARRAY.OF()
- ARRAY.ISARRAY()
- ARRAY.AT()
- ARRAY.COPYWITHIN()
- ARRAY.FLAT()
- ARRAY.FLATMAP()

JS


#30
Using JS
Array Methods
JS

Variable Scope

In JavaScript, **the scope of a variable determines where it can be used/applied within the code.** At its core, scope in JavaScript refers to the context or environment in which variables are declared and can be accessed. It dictates the visibility and lifetime of a variable, determining where in your code a particular variable is valid and accessible.

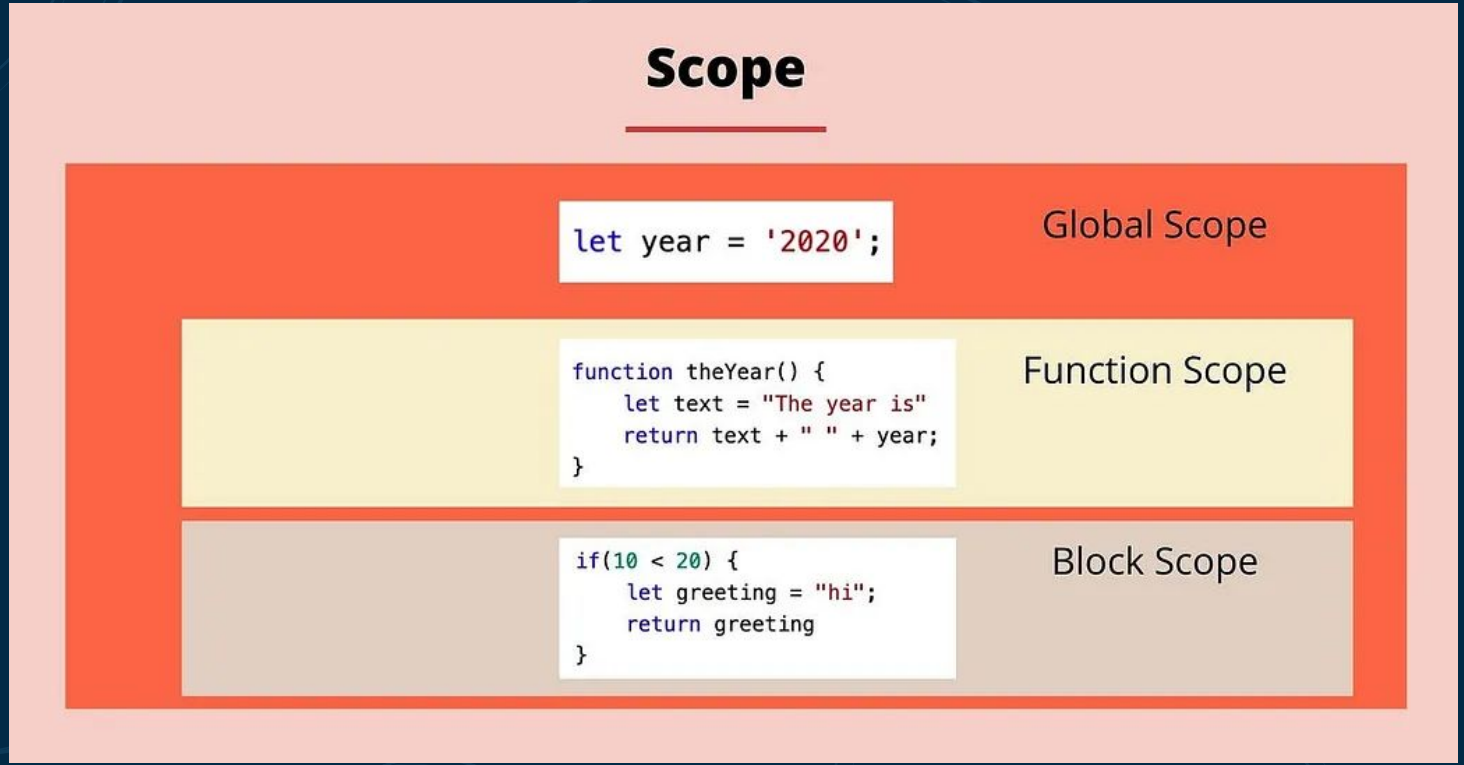
Variables can be declared in different scopes:

- **Global Scope:** In JavaScript, global scope is the widest scope available. Variables declared in global scope are accessible from anywhere in your code, whether it's inside functions, conditional statements, loops, or other blocks of code. Generally they are declared outside of all functions and blocks.

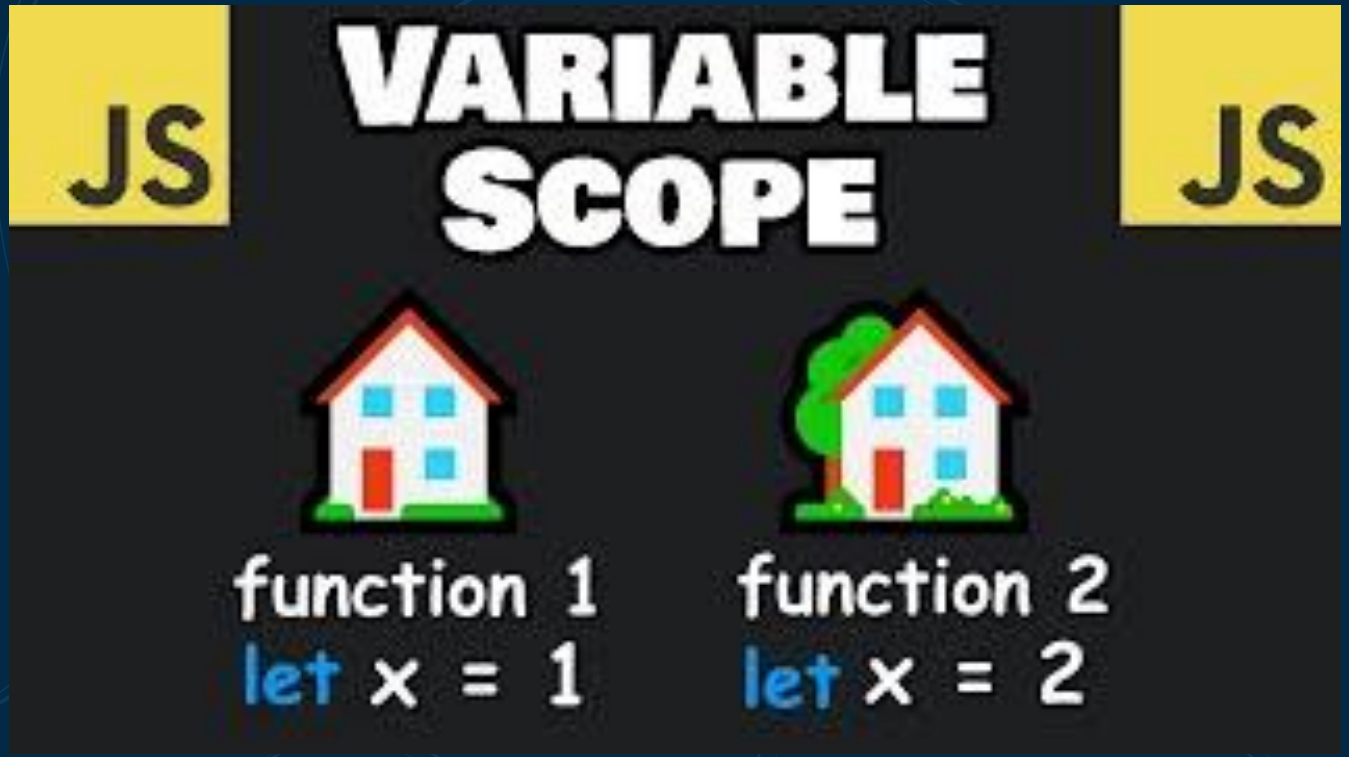
Variable Scope

- **Local (Function) Scope:** JavaScript has function scope: Each function creates a new scope. Variables declared within a JavaScript function, are LOCAL to the function. This means that any variables defined inside a function are not accessible (visible) from outside the function. Variables declared with var, let and const are quite similar when declared inside a function.
- **Block-Level Scope:** Block scope is created within specific code blocks, such as conditional statements (if, else, switch) and loops (for, while). Variables declared in block scope are confined to that block, offering a high degree of isolation.Variables declared with the var keyword can NOT have block scope. Variables declared inside a { } block cannot be accessed from outside the block.

Variable Scope



Variable Scope



Strict Equality Comparison ===

HTML DOM Methods

- The HTML DOM can be accessed with JavaScript (and with other programming languages).
- In the DOM, all HTML elements are defined as objects.
- The API is the properties and methods of each object.
- A property is a value that you can get or set (like changing the content of an HTML element).
- A method is a type of functions that allows you to take some action on your webpage (like add or deleting an HTML element).
- For example, you can use DOM methods to access HTML elements using id, attribute, tag name, class name, etc., add events to the document or HTML elements, add new HTML elements to the DOM, etc.

DOM HTMLCollection

- **An HTMLCollection is an array-like collection (list) of HTML elements.** (but it's not an array so arrays methods won't work)
- The elements in a collection can be accessed by index (starts at 0).
- The length Property returns the number of elements in the collection.
- The getElementsByTagName() Method and getElementsByClassName() Method - these are two methods that return such an HTML Collection
- You can loop through the list and refer to the elements with a number (just like an array). Typecast HTMLCollection to an array to be able to use array methods like forEach(), from(), etc.
- **The following properties and methods can be used on an HTMLCollection: .length, item(), namedItem().**

document.getElementsByTagName()

- **The document.getElementsByTagName() method returns all the element of specified tag name.**
- Syntax: document.getElementsByTagName("HTMLtag")
- Example: let *allparas* = *document*.getElementsByTagName("p");
- The getElementsByTagName() method returns a collection of all child elements with a given tag name.
- An HTMLCollection is an array-like collection (list) of HTML elements.
- The length Property returns the number of elements in the collection.
- The elements can be accessed by index (starts at 0).
- An HTMLCollection is live. It is automatically updated when the document is changed.

document.getElementsByClassName()

- **The getElementsByClassName() method returns a collection of elements with a specified class name(s).**
- The getElementsByClassName() method returns an HTMLCollection.
- Syntax: getElementsByClassName(“classname”);
- Eg1: let *infoparas* = *document*.getElementsByClassName("infopara");

  This will return all the elements that have the class name “infopara”
- Eg2: document.getElementsByClassName("red test");

  This will return all the elements that have both class names “red” and “test”
- Eg3: document.getElementById("main").getElementsByClassName("test");

  Get all elements that have a class of “test”, inside of an element that has the ID of “main”. So those elements will be children of <main>

Accessing a style object

- The Style object represents an individual style statement.
- The HTML DOM allows JavaScript to change the style of HTML elements. Minimize the use of DOM to change inline styles.
- Syntax: document.getElementById(*id*).style.*property* = *new style;*
- Eg. document.getElementById("mypara").style.color = “rgb(255, 255, 255)”; // this changes the inline style of the element
- Almost all of the CSS properties that exist can be modified through JavaScript using the style object.
- Modifying CSS through Javascript does not impact the Document Object Model (DOM) since it deals with stylistic modifications on existing web page elements. Any HTML element is targetable through Javascript CSS.

document.querySelector() DOM Method

- The **DOM method querySelector()** returns the first Element within the document that matches the specified CSS selector, or group of CSS selectors. If no matches are found, null is returned.
- To return all matches (not only the first), use the querySelectorAll() instead.
- The CSS Selector can be: an HTML tag, CSS class or id
- Syntax: document.querySelector(“CSS Selector”);

  Eg1: document.querySelector(“img”);

  Eg2: document.querySelector(“.myclass”);

  Eg3: document.querySelector(“#myid”);
- CSS pseudo-elements will never return any elements.

document.querySelectorAll()
DOM Traversal Method

- The querySelectorAll() method returns all elements that matches a CSS selector(s).
- It returns a NodeList which is like an array/ HTML Collection.
- querySelectorAll() returns a static (not live) NodeList representing a list of the document's elements that match the specified group of selectors.
- Syntax: querySelectorAll(selectors)
- Eg. 1: const matches = document.querySelectorAll("p");
- Eg. 2: let red_divs = document.querySelectorAll(“div.red”);
- The elements are in document order – that is, parents before children, earlier siblings before later siblings.

Homework: Continue the HTML + CSS portion of your LCA2 project. If design is still left, complete it and create a working prototype/storyboard.

W3Schools JavaScript Tutorial

JavaScript for Web | MDN

Reference

1. JavaScript Array Methods | WebReference
2. Arrays and Their Properties in JavaScript
3. JavaScript Variable Scope - Programiz
4. Scope in JavaScript. | by devtalib | Medium
5. JavaScript – Global vs Local vs Block Scope - FreeCodeCamp
6. Logical Operators - JavaScript.info
7. JavaScript while Loop - W3Schools
8. Document: getElementsByClassName() method - Web APIs | MDN
9. JavaScript HTML DOM Methods - W3Schools
10. DOM HTMLCollection Object - W3Schools

Reference

11. HTMLCollection | MDN
12. getElementsByTagName - Tpoint Tech
13. HTML DOM Element getElementsByTagName() Method - W3Schools
14. HTML DOM Document getElementsByClassName() Method
15. HTML DOM Style object - W3Schools (List of CSS properties that can be modified with the DOM)
16. Changing Element Styling with Javascript CSS | Udacity
17. Enhanced for loop for HTMLCollection Elements - Stack Overflow
18. HTML DOM Document querySelector() Method - W3Schools
19. JavaScript HTML DOM NodeList Object - W3Schools
20. querySelector() and querySelectorAll() DOM Methods JavaScript - FreeCodeCamp

Fonts & colors used

This presentation has been made using the following fonts:

Blinker
(https://fonts.google.com/specimen/Blinker)

Inconsolata
(https://fonts.google.com/specimen/Inconsolata)