Language Map for JavaScript by Steven Blaine

Variable Declaration

Is this language strongly typed or dynamically typed? Provide at least three examples (with different data types or keywords) of how variables are declared in this language.

JavaScript is considered a “dynamically” typed, “loosely” typed, “weakly” typed, or "untyped" language, meaning it is not required to specify what type of information will be stored in a variable in advance.

Some representative variable declarations:

- var muppet = “Kermit”;
- var tacoCount = 2;
- var b = true;

Data Types

List all of the data types (and ranges) supported by this language.

Data types in JavaScript consist of primitive values and objects:

- Primitive values (immutable data represented directly at the lowest level of the language):
  - Boolean type (true or false)
  - Null type (only value is null)
  - Undefined type (unassigned value)
  - Number type (a double-precision 64-bit binary format IEEE 754 value which may store floating-point numbers between 2^-1074 and 2^1024, but can only safely store integers in the range -(2^53 − 1) to 2^53 – 1)
  - BigInt type (a numeric primitive that can represent integers with arbitrary precision; can safely store and operate on large integers even beyond the safe integer limit for Numbers)
  - String type (immutable representations of textual data; a set of “elements” of 16-bit unsigned integer values)
  - Symbol type (unique and immutable primitive value; may be used as the key of an Object property)
- Objects (collections of properties which are data properties or accessor properties)

Selection Structures

Provide examples of all selection structures supported by this language (if, if else, etc.) ***Don’t just list them, show code samples of how each would look in a real program.***

“If” specifies a block of code to be executed, if a specified condition is true:

```
if (hour < 18) {
  greeting = “Good day”;
}
```

“Else” specifies a block of code to be executed, if the same condition is false:

```
if (hour < 18) {
  greeting = “Good day”;
} else {
  greeting = “Good evening”;
}
```

“Else if” specifies a new condition to test, if the first condition is false:

```
if (time < 10) {
  greeting = “Good morning”;
} else if (time < 20) {
  greeting = “Good day”;
} else {
  greeting = “Good evening”;
}
```

“Switch” specifies multiple alternative blocks of code to be executed:

```
witch (new Date().getDay()) {
  case 0:
    day = “Sunday”;
    break;
  case 1:
    day = “Monday”;
    break;
  case 2:
     day = “Tuesday”;
    break;
  case 3:
    day = “Wednesday”;
    break;
  case 4:
    day = “Thursday”;
    break;
  case 5:
    day = “Friday”;
    break;
  case 6:
    day = “Saturday”;
}
```

Repetition Structures

Provide examples of all repetition structures supported by this language (loops, etc.) ***Don’t just list them, show code samples of how each would look in a real program.***

JavaScript supports different kinds of loops:

- “for” - loops through a block of code a number of times
  ```
  for (let i = 0; i < 5; i++) {
    text += “The number is “ + i + “<br>”;
  }
  ```

- “for/in” - loops through the properties of an object
  ```
  const person = {fname:“John”, lname:“Doe”, age:25};
  ```

	let text = “”;
	for (let x in person) {
	  text += person[x];
	}
	
	• “for/of” - loops through the values of an iterable object
	const cars = [“Volvo”, “Porsche”, “Mercedes”];
	let text = “”;
	for (let x of cars) {
	  text += x;
	}
	
	• “while” - loops through a block of code while a specified condition is true
	while (i < 10) {
	  text += “The number is ” + i;
	  i++;
	}
	
	• “do/while” - also loops through a block of code while a specified condition is true and will always execute at least once
	do {
	  text += “The number is ” + i;
	  i++;
	}
	while (i < 10);

Arrays
If this language supports arrays, provide at least two examples of creating an array with a primitive or String data types (e.g. float, int, String, etc.)
	JavaScript does support arrays (numbered indexes (as opposed to objects, which are unnumbered indexes), and two creation examples are below.
	
	const fruits = [“Banana”, “Orange”, “Apple”, “Mango”];
	
	const points = new Array(40, 100, 1, 5, 25, 10);

Data Structures *If this language provides a standard set of data structures, provide a list of the data structures and their Big-Oh complexity.*	<u>Name</u> | <u>Insert</u> | <u>Access</u> | <u>Search</u> | <u>Delete</u> | <u>Comments</u> Array | O(n) | O(1) | O(n) | O(n) | Insertion to the end is O(1).

	HashMap	O(1)	O(1)	O(1)	O(1)	Rehashing might affect insertion time.
	Map (using Binary Search Tree)	O(log(n))	-	O(log(n))	O(log(n))	Implemented using Binary Search Tree
	Set (using HashMap)	O(1)	-	O(1)	O(1)	Set using a HashMap implementation.
	Set (using list)	O(n)	-	O(n)	O(n)	Implemented using Binary Search Tree
	Set (using Binary Search Tree)	O(log(n))	-	O(log(n))	O(log(n))	Implemented using Binary Search Tree
	Linked List (singly)	O(n)	-	O(n)	O(n)	Adding/Removing to the start of the list is O(1).
	Linked List (doubly)	O(n)	-	O(n)	O(n)	Adding/Deleting from the beginning/end is O(1). But, deleting/adding from the middle is O(n).
	Stack (array implementation)	O(1)	-	-	O(1)	Insert/delete is last-in, first-out (LIFO)

Queue (naïve array implementaion)	O(1)	-	-	O(n)	Remove (Array.shift) is *O(n)*
Queue (array implementation)	O(1)	-	-	O(1)	Worst time insert is O(n). However amortized is O(1)
Queue (list implementation)	O(1)	-	-	O(1)	Using Doubly Linked List with reference to the last element.

Objects

If this language support object-orientation, provide an example of how you would write a simple object with a default constructor and then how you would instantiate it
.

Unlike other object-oriented languages, JavaScript is a “prototype-based object-oriented language,” which means it doesn’t have classes; rather, it defines behaviors using the constructor function and then reuses it using the prototype. In other words, there are no classes in JavaScript, only objects.

An example of creating an object with a constructor function:

```
function vehicle(name,maker,engine){
    this.name = name;
    this.maker = maker;
    this.engine = engine;
}
let car  = new vehicle(“GT”,“BMW”,“1998cc”);
console.log(car.name);
console.log(car.maker);
console.log(car[“engine”]);
```

Creation using object literals:

```
let car = {
    name : “GT”,
    maker : “BMW”,
    engine : “1998cc”
};
console.log(car.name);
console.log(car[“maker”]);
```

Creation using Object.create() method:

	`const coder = {`
	`  isStudying : false,`
	`  printIntroduction : function(){`
	`    console.log(“My name is ${this.name}. Am I studying?: ${this.isStudying}”);`
	`  }`
	`};`
	`const me = Object.create(coder);`
	`me.name = “Bert”;`
	`me.isStudying = true;`
	`me.printIntroduction();`

Runtime Environment *What runtime environment does this language compile to? For example, Java compiles to the Java Virtual Machine.* *Do other languages also compile to this runtime?*	There are two JavaScript runtime environments: • The runtime environment of a browser (*e.g.*, Chrome, or Firefox); and • The Node runtime environment. Yes, other Web development languages, including HTML and CSS, can run in a browser.
Libraries/Frameworks *What are the popular libraries or frameworks used by programmers for this language? List at least three (3) and describe what they are used for.*	JavaScript libraries include: • jQuery, which is used to simplify HTML document manipulation and traversal, animation, event handling, and Ajax; • React.js, which is used for building user interfaces; and • D3.js, which is used for document manipulation (including visualization) based on data.
Domains *What industries or domains use this programming language? Provide specific examples of companies that use this language and what they use it for* ***. E.g. Company X uses C# for its line of business applications.***	JavaScript is used across many industries, from finance to marketing to entertainment. Specifies company users include: • Microsoft, which uses JavaScript in connection with its Edge browser and Azure cloud service; • Netflix using the language as part of its configuration for distributing entertainment content; and • Meta, which requires JavaScript for its Facebook social media platform to execute.