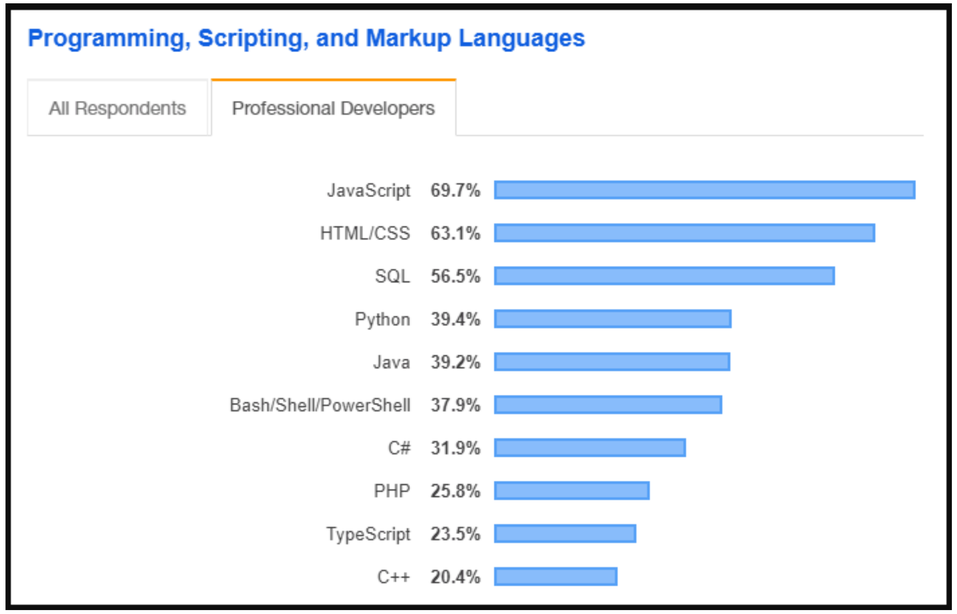
JAVASCRIPT FUNDAMENTALS

BASIC CONCEPTS

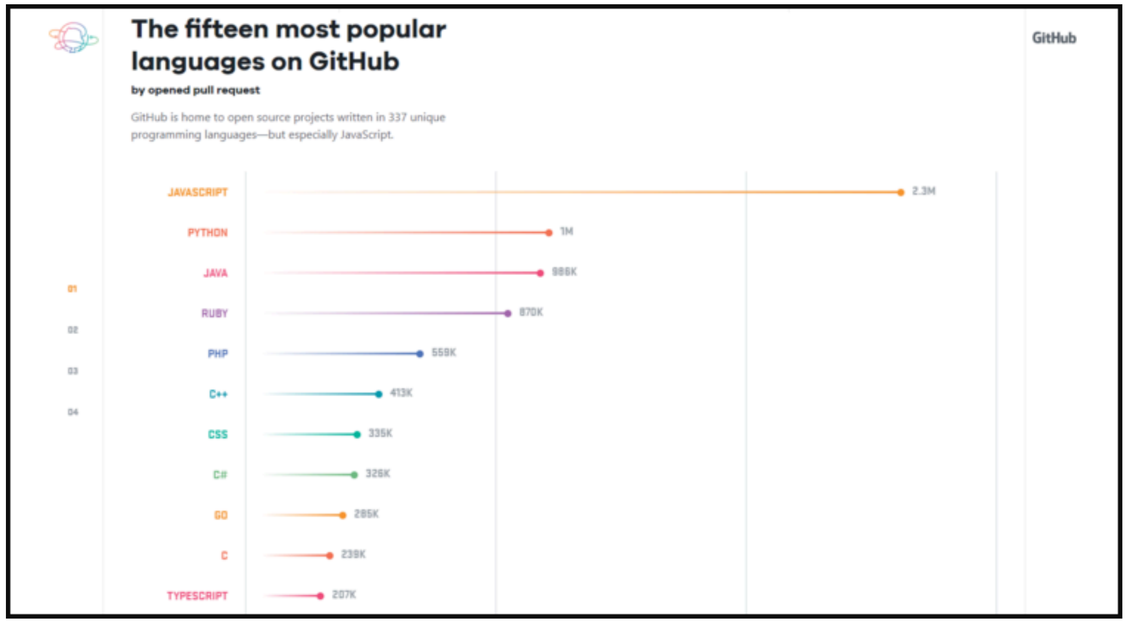
TABLE OF CONTENTS

- Introduce Javascript language
- Javascript environments
- Variables & data types
- Javascript Hoisting
- Basic operations
- Loop in Javascript
- Taking Condition
- Javascript function
- Working with Array
- Javascript Object

INTRODUCE JAVASCRIPT: THE MARKET

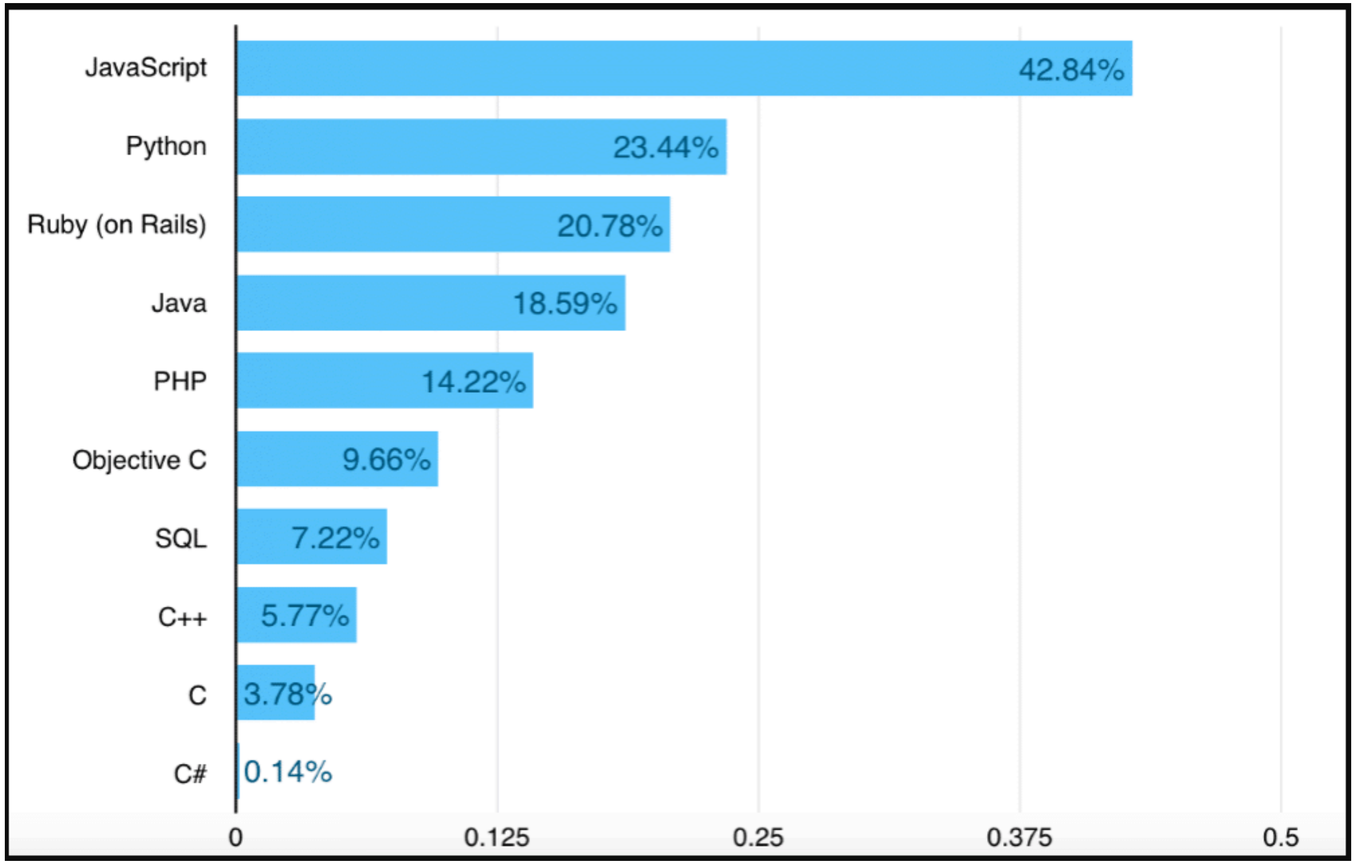


Top languages use by professional developers
Source: merehead.com



Top 15 languages on github
Source: simplytechnologies

INTRODUCE JAVASCRIPT: THE FASTEST LEARNING LANGUAGE



Language ranking by difficulty
Source: blockgeeks.com

INTRODUCE JAVASCRIPT: WHICH IS BEST WAY TO LEARN PROGRAMING LANGUAGES

- Master (or good enough) the 1st language before learning 2nd one.
- Each language has its own strengths.
  - C++
  - Java
  - C#
  - Python
- My advice for the 1st one is Javascript.

INTRODUCE JAVASCRIPT: LEARNING SOURCES & REFERENCES

- MDN - Mozilla Developer Network
- BLOG - Toidicodedao
- BLOG - The full snack
- Recommended books:
  - The clean code
  - The pragmatic programmer

JAVASCRIPT ENVIRONMENTS:

- Editors:
  - Notepad
  - Visual Code
  - Any text editor
- Runtime Environment:
  - Browser: Chome, Safari
  - NodeJS

SETUP ENVIRONMENT

VARIABLES & DATA TYPES:

- Declares a variable:
  - Keyword 'var'
  - Keyword 'let'
  - Keyword 'const'
- What's differents ?

- Available data types:
  - String
  - Number
  - Arrays
  - Object
  - Boolean
  - .....
  - What's primitive types ?

Learn more: https://www.w3schools.com/js/js_datatypes.asp

JAVASCRIPT HOISTING

- Variable Hoisting

```
console.log(counter); //result ?
var counter = 1;

var counter;
console.log(counter); //result ?
counter = 1;

counter = 1;
console.log(counter); //result ?
var counter;

counter = 1;
console.log(counter); //result ?
let counter;
```

=> All declaration using keyword 'var' will be moved to top

JAVASCRIPT HOISTING

- Function Hoisting

```
let x = 10;
let y = 20;

let result = add(x, y);
console.log(result);

function add(a, b) {
  return a + b;
}
```

=> All declaration using keyword 'var' will be moved to top

BASIC OPERATIONS: ARITHMETIC OPERATORS

Operator	Description
+	Adds two numeric operands.
-	Subtract right operand from left operand
*	Multiply two numeric operands.
/	Divide left operand by right operand.
%	Modulus operator. Returns remainder of two operands.
++	Increment operator. Increase operand value by one.
--	Decrement operator. Decrease value by one.

BASIC OPERATIONS: COMPARISION

Operators	Description
==	Compares the equality of two operands without considering type.
===	Compares equality of two operands with type.
!=	Compares inequality of two operands.
>	Checks whether left side value is greater than right side value. If yes then returns true otherwise false.
<	Checks whether left operand is less than right operand. If yes then returns true otherwise false.
>=	Checks whether left operand is greater than or equal to right operand. If yes then returns true otherwise false.
<=	Checks whether left operand is less than or equal to right operand. If yes then returns true otherwise false.

BASIC OPERATIONS: LOGICAL

Operator	Description
&&	&& is known as AND operator. It checks whether two operands are non-zero (0, false, undefined, null or "" are considered as zero), if yes then returns 1 otherwise 0.
||	|| is known as OR operator. It checks whether any one of the two operands is non-zero (0, false, undefined, null or "" is considered as zero).
!	! is known as NOT operator. It reverses the boolean result of the operand (or condition)

BASIC OPERATIONS: ASSIGNMENT

Assignment operators	Description
=	Assigns right operand value to left operand.
+=	Sums up left and right operand values and assign the result to the left operand.
-=	Subtract right operand value from left operand value and assign the result to the left operand.
*=	Multiply left and right operand values and assign the result to the left operand.
/=	Divide left operand value by right operand value and assign the result to the left operand.
%=	Get the modulus of left operand divide by right operand and assign resulted modulus to the left operand.

BASIC OPERATIONS: TERNARY

Syntax:

```
<condition> ? <value1> : <value2>;
```

Example: Ternary operator

```
var a = 10, b = 5;

var c = a > b? a : b; // value of c would be 10
var d = a > b? b : a; // value of d would be 5
```

LOOP IN JAVASCRIPT: DIFFERENT KINDS

- for
- for in
- for of
- while
- do while
- foreach

LOOP IN JAVASCRIPT: EXAMPLES

```
for (let step = 0; step < 5; step++) {
  // Runs 5 times, with values of step 0 through 4.
  console.log('Walking east one step');
}
```

```
let i = 0;
do {
  i += 1;
  console.log(i);
} while (i < 5);
```

```
let n = 0;
let x = 0;
while (n < 3) {
  n++;
  x += n;
}
```

TAKING CONDITION: IF...ELSE

```
let cheese = 'Cheddar';
if (cheese) {
  console.log('Yay! Cheese available for making cheese on toast.');
} else {
  console.log('No cheese on toast for you today.');
}
```

```
let x = 5;
let y = 4;
let z = 3;
let userName = Gemi;
if ((x === 5 || y > 3 || z <= 10) && (loggedIn || userName === 'Steve')) {
  console.log('codition is true');
}
else if {
  console.log('codition is false');
}
```

JAVASCRIPT FUNCTION:

```
function addSquares(a, b) {
   return (a * a) + (b * b);
}
a = addSquares(2, 3); // returns 13
b = addSquares(3, 4); // returns 25
c = addSquares(4, 5); // returns 41

=> Normal functions
```

```
function addSquares(a, b) {
  function square(x) {
    return x * x;
  }
  return square(a) + square(b);
}
a = addSquares(2, 3);
b = addSquares(3, 4);
c = addSquares(4, 5);

=> Nested functions
```

WORKING WITH ARRAY: WHAT'S ARRAY ?

- An array is a collection of elements.
- Store multiple values in a single variable.
- Arrays in JavaScript are not a type on their own.
- Arrays are objects.

WORKING WITH ARRAY: DECLARATION

- Declare a array in Javascript

```
var cars = ["Saab", "Volvo", "BMW"];
var person = ["John", "Doe", 46];
```

- Array element can be an object

```
myArray[0] = Date.now;
myArray[1] = myFunction;
myArray[2] = myCars;
```

- Accessing Javascript array elements

```
let mountains = ['Everest', 'Fuji', 'Nanga Parbat'];
console.log(mountains[0]); // 'Everest'
console.log(mountains[1]); // 'Fuji'
console.log(mountains[2]); // 'Nanga Parbat'
```

WORKING WITH ARRAY: BASIC OPERATIONS

- Array - length

```
let arr = [0, 1, 2];
arr[4] = "new";
console.log(arr) // [0, 1, 2, empty, "new"]
console.log(arr[3]) // undefined
arr.length = 10;
console.log(arr) // [0, 1, 2, empty, "new", empty × 5]
arr.length = 2; console.log(arr) // [0, 1]
```

- Array - add items to end of array

```
let fruits = ['Apple', 'Banana'];
let newLength = fruits.push('Orange');
```

- Array - add to beginning of array

```
let fruits = ['Apple', 'Banana'];
let newLength = fruits.unshift('Orange');
```

WORKING WITH ARRAY: BASIC OPERATIONS

- Remove an item from the end

```
let fruits = ['Apple', 'Banana'];
let last = fruits.pop();
```

- Remove an item from beginning

```
let fruits = ['Apple', 'Banana'];
let first = fruits.shirt();
```

- Remove an item by index position

```
let fruits = ['Apple', 'Banana', 'Orange'];
let removedItem = fruits.splice(1,1);
```

WORKING WITH ARRAY: BASIC OPERATIONS

- Find item index

```
let fruits = ['Apple', 'Banana', 'Orange'];
let index = fruits.indexOf('Banana');
```

- Copy an arrays

```
let fruits = ['Apple', 'Banana'];
let shallowCopy = fruits.slice();
```

- For each array element

```
let fruits = ['Apple', 'Banana', 'Orange'];
fruits.forEach(function(item, index, array) {
   console.log(item, index);
});
```

WORKING WITH ARRAY: BASIC OPERATIONS

- Map

```
const array1 = [1, 4, 9, 16];
const map1 = array1.map(function(item){
    return item * 2;
});
console.log(map1)
//output: [2, 8, 8, 32]
```

- Join 2 arrays

```
let arr1 = [1, 2, 3]
let arr2 = ["a", "b", "c"];
let arr3 = arr1.concat(arr2);
let arr4 = arr1.concat(arr2, arr3)
console.log(arr4);
//[1, 2, 3, "a", "b", "c", 1, 2, 3, "a", "b", "c"]
```

WORKING WITH ARRAY: BASIC OPERATIONS

- Filter

```
const array1 = [1, 4, 9, 16];
const result = array1.filter(function(item){
    return item % 2 == 0;
});
console.log(result)
//output: [4, 16]
```

- Reduce

```
const numbers = [1, 2, 3, 4, 5];
const sum = numbers.reduce(function(accumulator, currentValue){
    return accumulator + currentValue;
});
console.log(sum)
//output: [4, 16]
```

JAVASCRIPT OBJECT:

- Use to store data : key => value
- Almost objects in Javascript are instances of Object
- Objects are variables too

Object	Properties	Methods
	car.name = Fiat	car.start()
	car.model = 500	car.drive()
	car.weight = 850kg	car.brake()
	car.color = white	car.stop()

JAVASCRIPT OBJECT:

- Define an object

```
var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};
```

- Constructor pattern

```
function Person(firstName, lastName) {
  this.firstName = firstName;
  this.lastName = lastName;
}
var psn = new Person('Hoang', 'Pham')
//iterate over an object
for (let prop in person) {
  console.log(prop);
}
```

JAVASCRIPT OBJECT:

- Object.create()
  Creates a new object with the specified prototype object and properties
- Object.assign()
  Copies the values of all enumerable own properties from one or more source objects to a target object.
- Object.keys()
  Returns an array containing the names of all of the given object's own enumerable string properties.
- Object.values()
  Returns an array containing the values that correspond to all of a given object's own enumerable string properties.
- Object.freeze()
  Freezes an object. Other code cannot delete or change its properties.

ASSIGNMENT

EXERCISE 1

1. Write a function to format money string (see examples):
   10000000 => “10,000,000" || 123456 => “123,456" || 12000.02 => "12,000.02"

2. Write a function for format money in shorten (see examples):
   1000000000 => 1B, 1000000 => 1M
   1000 => 1K, 1123400000 => 1.12B , 1342222 => 1.34M

3. Write the function to count how many words appear in a string:
   oneTwoThree => 3

4. Write the function get the get the Extension of file:
   “image.png” => “png”. “Sound.mp3” => “mp3”

EXERCISE 2

1. Write the function to calculate the factorial of a number.

2. Write the function to get a random integer between 2 numbers: min , max;

3. Write the function get a random element from an arrays.

3. Given two arrays of integers, find which elements in the second array are missing from the first array.

Example:

arr = [7,2,5,3,5,3]
brr = [7,2,5,4,6,3,5,3] => [4,6]

EXERCISE 3: 2D ARRAYS

We have a rectangle garden with 3 row and 5 column, each cell had a bomb or no bomd, (0: SAFE, 1: BOMB)

problem: find all the safe way to go from the left to the right of the garden.

Input: [[0,1,1], [0,1,1], [0,1,1], [0,1,1], [0,0,1]]

Output : [[0,0,0,0,0] , [0,0,0,0,1]