**Introduction to Programming**

* it is simply giving computers a set of instructions to complete a task
* A computer program is just a series of instructions based on some rules.
* These instructions tell a computer what to do under some specific circumstances and they are written to perform a specific set of tasks.

**Programming Language**

* A programming language is sort of like a bridge that we use to facilitate communication between computers and people.

**Memory, Binary code, Machine Language, Low Level Language, CPU**

* For a computer to understand a programming language, it needs to save it somewhere in its random access memory. RAM or on a disk in binary code, which is represented by a series of zeros and ones. The binary code is a representation of machine language. That a computer can understand all of the zeros and ones in binary language represent electrical switches. Finally, a computer's central processing unit, or CPU can comprehend these binary electrical signals and update itself accordingly. This machine language, with its series of zeros and ones, is known as a low level language because it's closer to being understood by a computer's CPU.

**High Level Language, Interpreting language**

* high level languages such as JavaScript, that needs to be interpreted, that is converted to binary code that a CPU will be able to work with. So we might think of JavaScript and other high level languages as a tool that helps us to communicate with the CPU.

**Why Javascript?**

* Inventor Brendan Eich
* JavaScript is a language that builds interactivity into web pages.
* It is literally the language of the web
* JavaScript has been the main way to interact with web pages on the client side, the front end side of websites and web applications.
* Using JavaScript updates are displayed in real time on our devices. Some common examples include interactive maps and client side form validation.
* some alternatives to JavaScript, such as VBScript and more recently TypeScript. But even typescript compiles down to JavaScript so browsers can understand it.
* JavaScript is currently the only computer language that allows us to directly interact with our web pages dynamically on the client. It's baked into the browser.
* Every browser has Javascript engine
* JavaScript was a simple scripting language <Script> HTML element
* JavaScript as a language is not a completely separate, stand-alone entity. It only exists as an implementation. This implementation is known as a JavaScript engine.
* the only environment in which it was possible to run a JavaScript engine, was the browser.
* developers write JavaScript code, they are using it to interact with a JavaScript engine.
* the JavaScript engine itself comes with different ways to interact with various other parts of the browser. These are known as Browser APIs.
* the code that you write in the JavaScript programming language allows you to: 1. Interact with the JavaScript engine inside of the browser 2. Interact with other browser functionality that exists outside of the JavaScript engine, but is still inside the browser.
* traditionally it was possible to interact with the JavaScript engine only inside of the browser, this all changed in 2009, when Node.js was built by Ryan Dahl.
* He came up with a way to use a JavaScript engine as a stand-alone entity. Suddenly, it was possible to use JavaScript outside of the browser, as a separate program on the command line, or as a server-side environment.
* Today, JavaScript is ubiquitous and is running in browsers, on servers, actually, on any device that can run a JavaScript engine.

**Programming in Javascript**

* Javascript is used in many scenarios, for example, in the browser to help add various behaviors and interactivity, like adding an item to a shopping cart when you click a button.
* On the server, it can be used to power up websites, communicate with databases, and provide a native fields to web apps.
* It's used to build mobile apps using technologies like React Native and it's used to program devices known as the Internet of Things. In a nutshell, JavaScript is everywhere.

**Browser Vendors, Incompatible problem and JQuery**

* different companies built Internet browsers and were referred to as browser vendors. with different browsers, different behaviors with various discrepancies between other browsers. This resulted in developers sometimes having to write separate JavaScript code for different browsers. This wasn't the best use of developer time and could lead to a frustrating experience for end-users. Out of this frustration, several projects appeared trying to solve this compatibility problem. One such project managed to solve these issues and it was a library named jQuery. Using this library, all a developer needed was to import jQuery and write code using its features. This code would then work across all browsers. At the time, it was like magic, as it was such a great solution to the problem of incompatibility and jQuery became the most popular JavaScript library for well over a decade. Recently React, solved many of the issues associated with creating, updating and maintaining complex websites. Some other frameworks to solve the same problem Knockout, Backbone, Angular, Ember, Vue, Alpine, and others.

**Comments, Semicolon in Javascript**

* Single line, Multi line
* //this is single line comment
* /\* Multiline comments \*/
* Each line ends with semicolon. Optional. Browser handles it even there is no semicolon

**Developer tool - console**

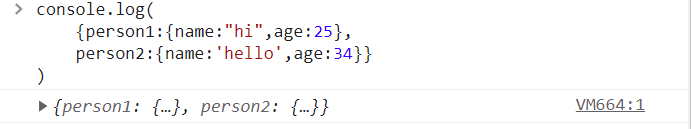
* **ESC** key will toggle on and off the console regardless of the currently active Developer Tools panel
* to run single line of code Enter key
* To type multiple lines of code before you run them press SHIFT + ENTER to get onto next line
* **console.log("%cHello, World", "color: blue; font-size: 40px");**
* If you add the **%c** right after the **"** character, you can then style the console output by adding the **,** character after the second **"**, and then, inside another pair of **"** and **"** characters, use valid CSS code to style the words you want to output in the console.
* "concatenation operator" - **console.log("Hello " + "there, " + "World")**outputting three separate pieces of text using the , character instead:

console.log("Hello ", "there, ", "World") both same output Hello there, World.

**Variables**

* Name given to a value
* Variable names are case-sensitive, so myVar, MyVar, and myvar are all different
* The variable’s memory location is indicated by the memory address.
* var, let, const





**Data types**

* primitive data types:  **string, number, Boolean, null** (absence of value, variable does not contain a value), **undefined** which can hold only the value undefined usually refers to a variable that has not been assigned a value (variable does not contain a value), ES6 new primitive date type **BigInt** data type that accommodates a greater range of numbers than number data type  **and symbol** data type used as a unique identifier
* Data : each piece of information in your app has a different value and all values are collectively referred to as data

**Operators**

* An operator is used to manipulate individual data items and return a result.
* +, -, /, \*, =, >,<,==,!=, &&, ||, !

**Numbers**

* represents integer and decimal points
* 10\*\*2 =100, % modulus operator

**Strings**

* Collection of characters enclosed in single or double quotes
* “It’s a lovely day” - correct
* 'It\'s a lovely day' output: "It's a lovely day" - to escape ‘ add \

**Booleans**

* True or false
* == equality operator, === strict equality operator
* 100 == “100” true. Compare only the value not datatype
* 100 === “100” false
* 1 != 1 false
* 1 !== “1” true
* 1 + “2” = 12
* 5 > 4 > 3; // the 5 > 4 is evaluated first (to `true`), then true > 3 is evaluated to `false`, because the `true` value is coerced to `1`

**Conditions and Loops**

**IF & Switch**

* If (condition == true) { // execute code } else { //execute code }
* flow control statements. In other words, they control the flow of execution of your code, so that some code can be skipped, while other code can be executed.
* Switch, case, default

| var age = 0;  if(age>=65)  console.log("You get your income from your pension");  else if(age<65 && age>=18)  console.log("Each month you get a salary");  else if(age<18)  console.log("You get an allowance");  else  console.log("The value of the age variable is not numerical");  var day = "Monday";  switch(day){  case 'Monday':  console.log('vaccuming');  break;  case 'Tuesday':  console.log('laundry')  break;  case 'Wednesday':  console.log('Moping')  break;  case 'Thursday':  console.log('Walking')  Break;  case 'Friday':  console.log('Yoga')  break;  case 'Saturday':  console.log('Temple visit')  break;  case 'Sunday':  console.log('Restaurant');  break;  default:  console.log("enter valid day");  break;  } |
| --- |

**Loops**

* JavaScript developers use loops to continually execute repeated blocks of code until a certain condition is satisfied.
* Code that repeats as long as specified condition is true
* Infinite loop
* for(var i =1; i<101;i++) { console.log(i); }
* FOR loop, While loop, NESTED loop
* Var i =1; while( i < 4 ) { console.log(i); i = 1 + 1; }

**Nested Loops**

* The more nested loops there are, the slower your code will run.
* Example Multiplication table

| //nested loops - one inside another  for (var firstNum = 0; firstNum < 2; firstNum++) {  for (var secondNum = 0; secondNum < 10; secondNum++) {  console.log(firstNum + " times " + secondNum + " equals " + firstNum \* secondNum);  }  }  Output:  The output now is:  0 times 0 equals 0  0 times 1 equals 0  0 times 2 equals 0  0 times 3 equals 0  0 times 4 equals 0  0 times 5 equals 0  0 times 6 equals 0  0 times 7 equals 0  0 times 8 equals 0  0 times 9 equals 0  1 times 0 equals 0  1 times 1 equals 1  1 times 2 equals 2  1 times 3 equals 3  1 times 4 equals 4  1 times 5 equals 5  1 times 6 equals 6  1 times 7 equals 7  1 times 8 equals 8  1 times 9 equals 9 |
| --- |

| //nested loops - one inside another  for (var i = 100; i > 10; i = i - 10) {  for (var j = 10; j > 4; j = j - 5) {  console.log(i + " divided by " + j + " equals " + i / j);  }  }  Output:  100 divided by 10 equals 10  100 divided by 5 equals 20  90 divided by 10 equals 9  90 divided by 5 equals 18  80 divided by 10 equals 8  80 divided by 5 equals 16  70 divided by 10 equals 7  70 divided by 5 equals 14  60 divided by 10 equals 6  60 divided by 5 equals 12  50 divided by 10 equals 5  50 divided by 5 equals 10  40 divided by 10 equals 4  40 divided by 5 equals 8  30 divided by 10 equals 3  30 divided by 5 equals 6  20 divided by 10 equals 2  20 divided by 5 equals 4 |
| --- |

var cubes = 'ABCDEFG';

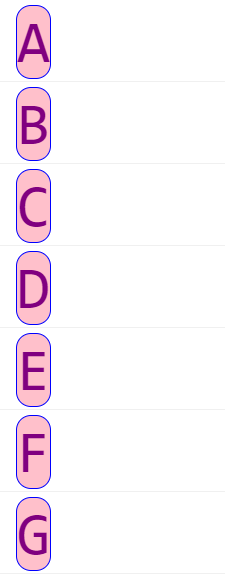
//styling console output using CSS with a %c format specifier

for (var i = 0; i < cubes.length; i++) {

var styles = "font-size: 40px; border-radius: 10px; border: 1px solid blue; background: pink; color: purple";

console.log("%c" + cubes[i], styles)

}

Output: 

**Knowledge Check**

**Question 1**

You can run JavaScript in a web browser's devtools console.

True or false

True Correct

That's correct! The devtools console is useful for running JavaScript code.

**Question 2**

Which of the following are valid comments in JavaScript? Select all that apply.

\ Comment 1

// Comment 2

Correct

That's correct! // is used for inline comments.

##

## Comment 3

##

/\*

\* Comment 4

\*/

Correct

That's correct! /\* and \*/ are define the beginning and end of multi-line comments.

**Question 3**

Which of the following are valid data types in JavaScript? Select all that apply.

1 / 1 point

string

Correct

That's correct! The string data type represents a sequence of characters in JavaScript.

numbers

Correct

That's correct! Numbers represent both integer and decimal point numeric values.

booleans

Correct

That's correct! The boolean data type has one of two values; true or false.

null

Correct

That's correct! The null data type represents the absense of a value.

4.

Question 4

Which of the following is the logical AND operator in JavaScript?

1 / 1 point

1

2

3

&

1

2

3

&&

1

2

||

1

2

3

|\

Correct

That's correct! && is the logical AND operator used for condition checks.

5.

Question 5

Which of the following is the assignment operator in JavaScript?

1 / 1 point

1

2

3

=

1

2

3

==

1

2

3

===

Correct

That's correct! The = symbol is used to assign to variables in JavaScript.

6.

Question 6

How many times will the following code print the word 'Hello'?

1

2

3

4

5

for(var i = 0; i <= 5; i++) {

console.log("Hello");

}

1 / 1 point

4

5

6

Correct

That's correct! 'i' starts with the value '0'. The condition checks if 'i' is less than or equal to '5'. Each loop increments 'i' by '1'. This means that the loop will run 6 times.

7.

Question 7

What will print out when the following code runs?

1

2

3

4

5

6

7

8

9

10

var i = 3;

var j = 5;

if(i == 3 && j < 5) {

console.log("Hello");

} else {

console.log("Goodbye");

}

1 / 1 point

Hello

Goodbye

Nothing

Correct

That's correct! The condition checks if 'i' is equal to '3' AND if 'j' is less than '5'. Since the result of this condition is false, the code inside the else statement will run.

8.

Question 8

What will print out when the following code runs?

1

2

3

4

5

6

7

8

9

10

var i = 7;

var j = 2;

if(i < 7 || j < 5) {

console.log("Hello");

} else {

console.log("Goodbye");

}

1 / 1 point

Hello

Goodbye

Nothing

Correct

That's correct! The condition checks if 'i' is less than '7' OR if 'j' is less than '5'. Since the result of this condition is true, the code inside the if statement will run.

9.

Question 9

The result of !false is:

1 / 1 point

true

undefined

Correct

Correct! When you add the NOT operator before a boolean value, the returned value is the opposite of the boolean value. Thus, !false is evaluated to true, and !true is evaluated to false.

10.

Question 10

What does the operator symbol || represent in JavaScript?

1 / 1 point

The logical AND operator

The logical NOT operator

The logical OR operator

Correct

That's correct. In JavaScript, the || is the logical OR operator.