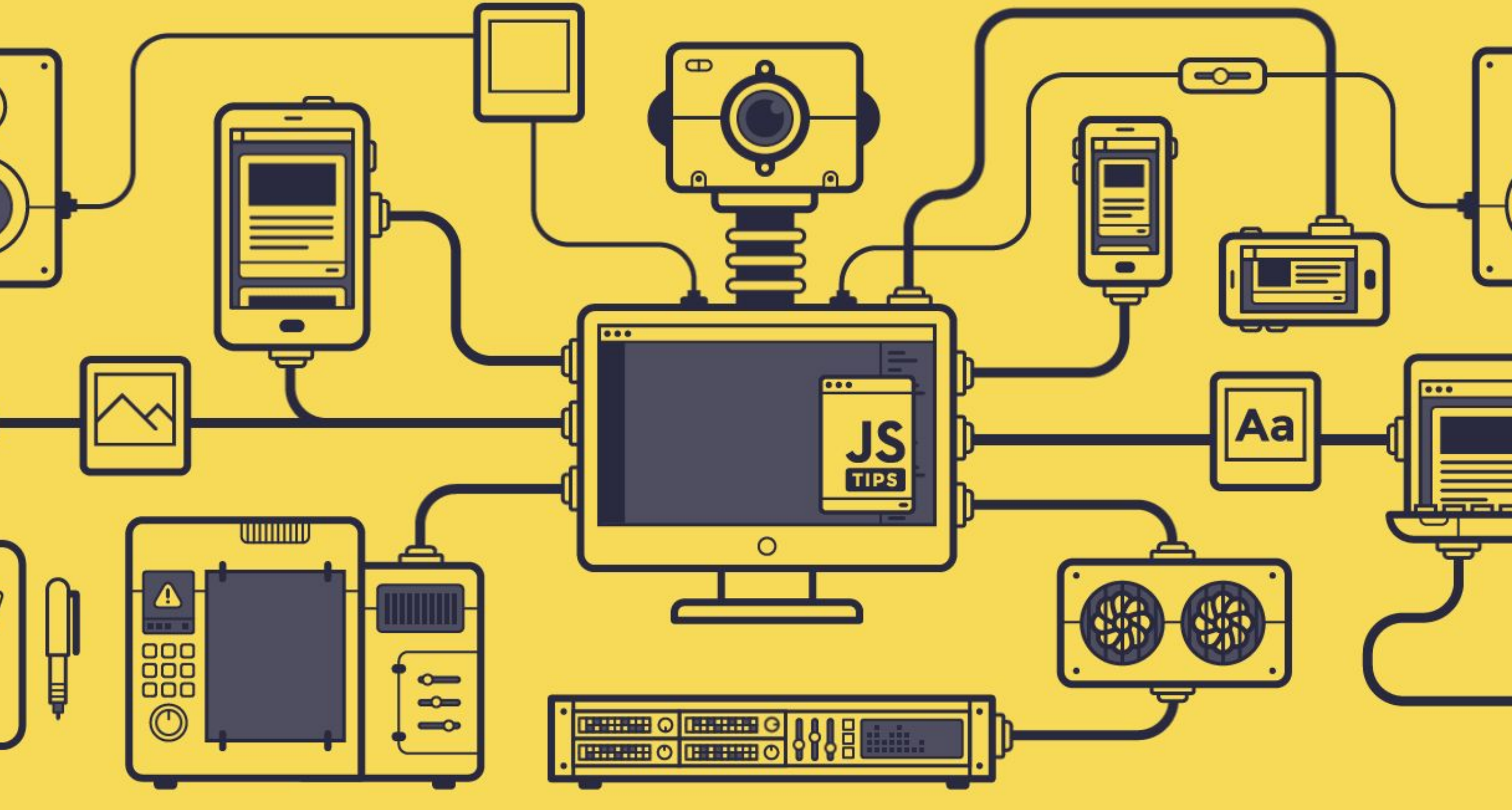
JavaScript

A powerful programming language that can add interactivity to a website.


JS
TIPS
Aa

Contents

1. Introduction

JavaScript was invented by **Brendan Eich** in 1995.

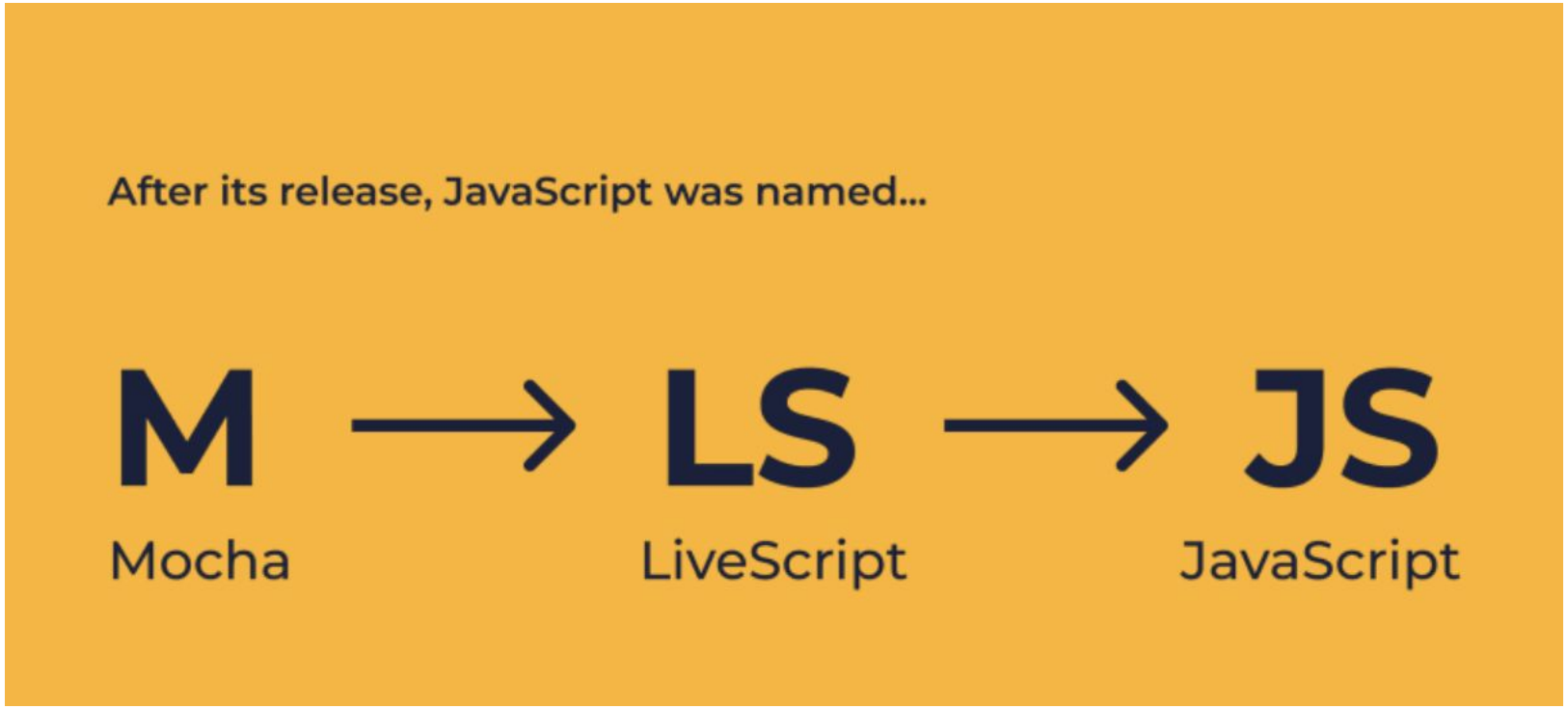
It was developed for **Netscape 2**, and became the **ECMA-262** standard in 1997.

After Netscape handed JavaScript over to ECMA (originally the European Computer Manufacturers Association, now just Ecma), the Mozilla foundation continued to develop JavaScript for the Firefox browser.

HTML and **CSS** give a website structure and style, JavaScript lets you add functionality and behaviors to your website. This allows visitors to interact with your website in various creative ways.

1. Introduction

In September 1995, a Netscape programmer named Brendan Eich developed a new scripting language in just 10 days. It was originally called Mocha



1. Introduction

- **What Is ECMAScript?**

When **JavaScript** was first introduced by **Netscape**, there was a war going on between all the browser vendors on the market at the time. Microsoft and several other browser vendors **implemented** their own **versions of JavaScript** (with different names and syntax) in their respective browsers. This created a huge headache for **developers**, as code that worked fine on one browser was a total waste on another.

1. Introduction

As a result, Netscape submitted JavaScript to the **European Computer Manufacturers Association** (ECMA) for standardization in order to ensure proper maintenance and support of the language. Since JavaScript was standardized by ECMA, it was officially named **ECMAScript**.

1. Introduction

Year	ECMA	Browser
1995		JavaScript was invented by Brendan Eich
1996		Netscape 2 was released with JavaScript 1.0
1997		JavaScript became an ECMA standard (ECMA-262)
1997	ES1	ECMAScript 1 was released
1997	ES1	IE 4 was the first browser to support ES1
1998	ES2	ECMAScript 2 was released
1998		Netscape 42 was released with JavaScript 1.3
1999	ES2	IE 5 was the first browser to support ES2
1999	ES3	ECMAScript 3 was released
2000	ES3	IE 5.5 was the first browser to support ES3
2000		Netscape 62 was released with JavaScript 1.5
2000		Firefox 1 was released with JavaScript 1.5
2008	ES4	ECMAScript 4 was abandoned

1. Introduction

2015	ES6	ECMAScript 6 was released
2016	ES6	Full support for ES6 in Chrome 51
2016	ES6	Full support for ES6 in Opera 38
2016	ES6	Full support for ES6 in Edge 14
2016	ES6	Full support for ES6 in Safari 10
2015	ES6	Full support for ES6 in Firefox 52
2018	ES6	Full support for ES6 in all browsers **

1. Introduction

Before JavaScript was primarily used in:

- making web pages interactive

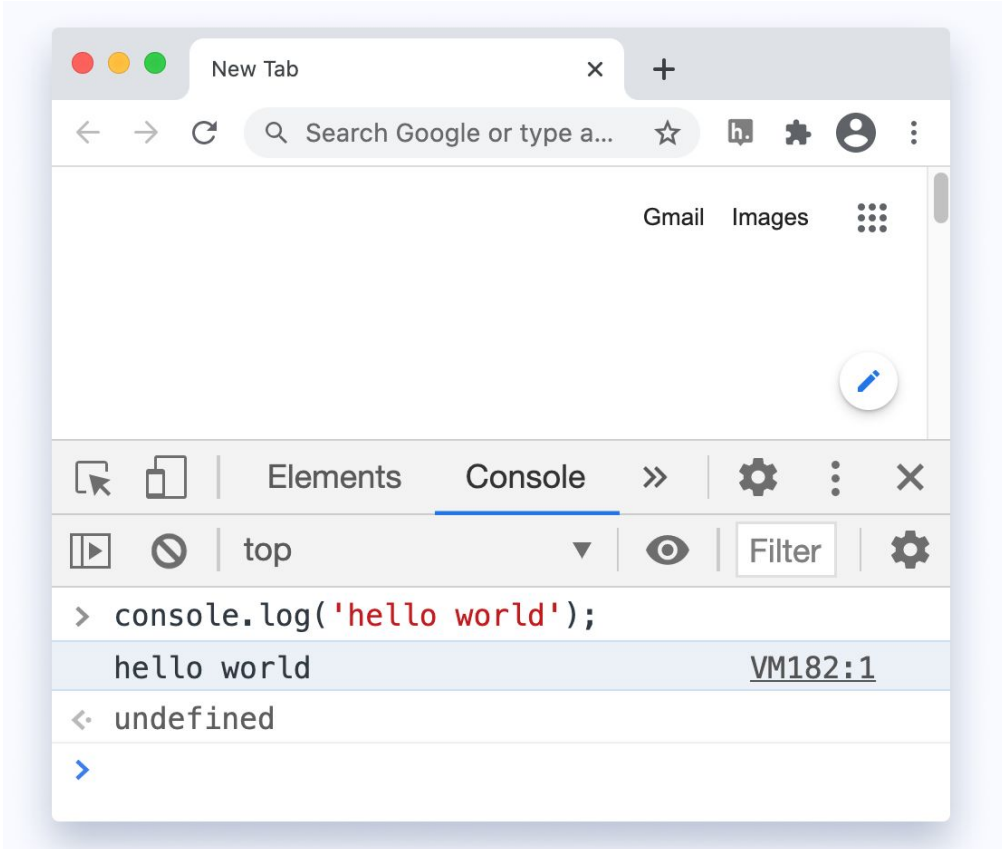
Nowadays, JavaScript is also used in:

- server-side applications
- game development
- mobile apps
- web apps

1. Introduction

you can run **JavaScript** in several ways:

- Using console tab of web browsers
- Using Node.js
- By creating web pages



2. Using Console Tab of Web Browsers

All the popular web browsers have built-in JavaScript engines.

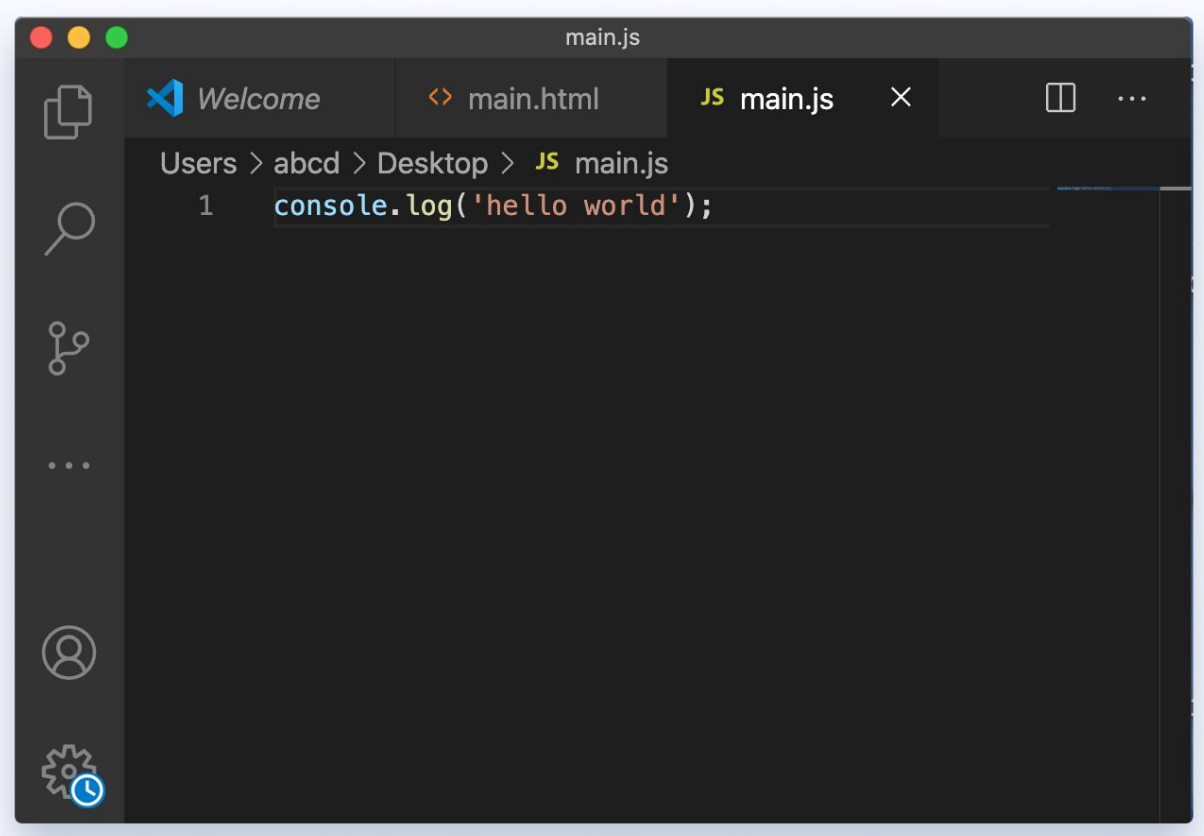
1. Introduction

2. Using Node.js

Node is a back-end run-time environment for executing JavaScript code. To run **JS** using **Node.js**, follow these steps:

- Install the latest version of Node.js.
- Install an IDE/Text Editor like Visual Studio Code. In VS code, create a file > write JS code > save it with **.js** extension.

1. Introduction



Hit the command: $**node hello.js**

1. Introduction

3. By Creating Web Pages

```
<!DOCTYPE html>
    <html lang="en">
        <head>
            <meta charset="UTF-8">
            <script src="./index.js"></script>
        </head>
    <body>
        <header>
            <nav></nav>
        </header>
    </body>
</html>
```

2. Variable and Constant

In programming, a variable is a container (storage area) to hold data. For example,

```
let num = 5;
```

Here, **num** is a variable. It's storing 5.

Declare Variables

In JavaScript, we use either **var** or **let** keyword to declare variables. For example,

```
var x;
let y;
```

2. Variable and Constant

Var vs Let

var	let
`var` is used in the older versions of JavaScript	`let` is the new way of declaring variables starting **ES6 (ES2015)**.
`var` is function scoped (will be discussed in later tutorials).	`let` is block scoped (will be discussed in later tutorials).
For example, `var x;`	For example, `let y;`

Note: It is recommended we use let instead of var. However, there are a few browsers that do not support let

2. Variable and Constant

Initialize Variables

We use the assignment operator = to assign a value to a variable.

```
let x;
x = 5;
```

You can also initialize variables during its declaration.

```
let x = 5;
let y = 6;
```

If you use a variable without initializing it, it will have an undefined value.

```
let x; // x is the name of the variable
console.log(x); // undefined
```

2. Variable and Constant

Rules for Naming JavaScript Variables

1. Variable names must start with either a letter, an underscore _, or the dollar sign $. For example,

```
//valid
let a = 'hello';
let _a = 'hello';
let $a = 'hello';
```

2. Variable names cannot start with numbers. For example

```
//invalid
let 1a = 'hello'; // this gives an error
```

2. Variable and Constant

3. JavaScript is case-sensitive. So y and Y are different variables. For example,

```
let y = "hi";
let Y = 5;
console.log(y); // hi
console.log(Y); // 5
```

4. Keywords cannot be used as variable names. For example

```
//invalid
let new = 5; // Error! new is a keyword.
```

2. Variable and Constant

Notes:

- Though you can name variables in any way you want, it's a good practice to give a descriptive variable name. If you are using a variable to store the number of apples, it better to use `apples` or `numberOfApples` rather than `x` or `n`.
- In JavaScript, the variable names are generally written in camelCase if it has multiple words. For example, `firstName`, `annualSalary`, etc.

2. Variable and Constant

Constants

The const keyword was also introduced in the **ES6**(ES2015) version to create constants.

```
const x = 5;
```

Once a constant is initialized, we cannot change its value.

```
const x = 5;
x = 10; // Error! constant cannot be changed.
console.log(x)
```

Also, you cannot declare a constant without initializing it.

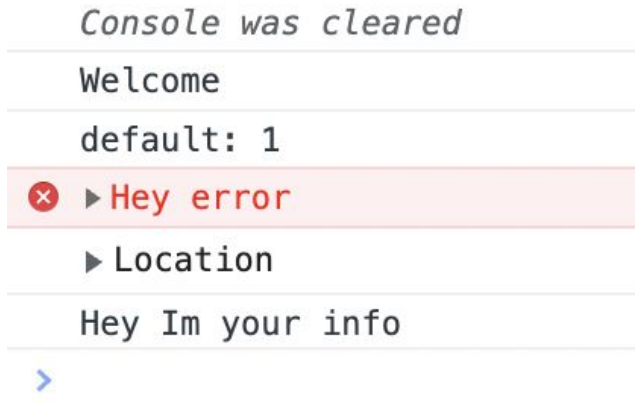
3. Console

All modern browsers have a web console for debugging. The **console.log()** method is used to write messages to these consoles. For example,

```
let sum = 44;
console.log(sum); // 44
```

The console object can be accessed from any global object. It's exposed as Window.console, and can be referenced as **console**. It has many methods such as:

```
console.clear()
console.log("Welcome")
console.count()
console.error("Hey error")
console.dir(window.location)
console.info("Hey Im your info")
```



4. Comment

- JavaScript comments can be used to explain JavaScript code, and to make it more readable.
- Comments can also be used to prevent execution, when testing alternative code.
- There are 2 types of comment.
  - Single Line Comments (//)
  - Multi-line comment (/* ... */)

- **Note:**
  - It is most common to use single line comments
  - Block comments are often used for formal documentation.

5. Output

Rendering User Interfaces

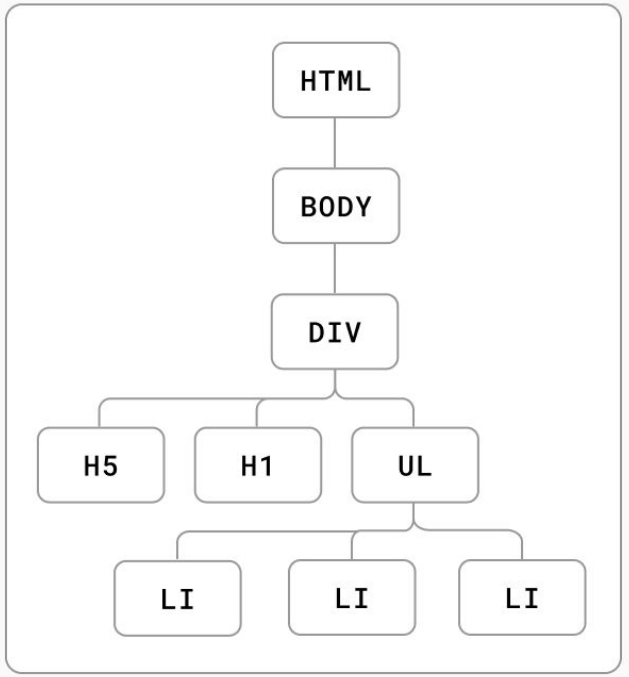
When a user visits a web page, the server returns an HTML file to the browser that may look like this:

HTML

index.html

```
1  <html>
2    <body>
3      <div>
4        <h5>Meet the</h5>
5        <h1>Engineers</h1>
6        <ul>
7          <li>A. Lovelace</li>
8          <li>G. Hopper</li>
9          <li>M. Hamilton</li>
10       </ul>
11     </div>
12   </body>
14 </html>
15
16
```

DOM



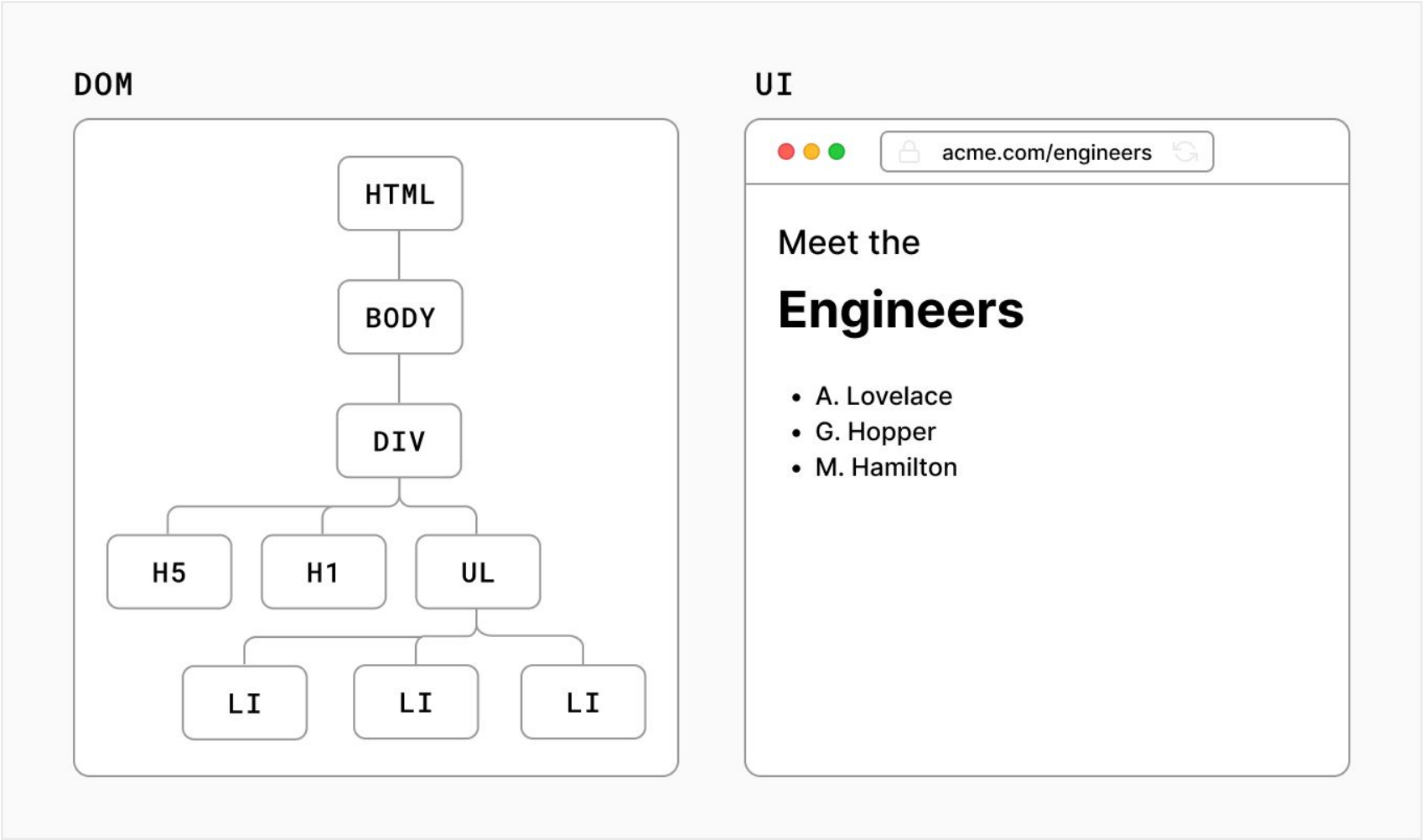
5. Output

The browser then reads the HTML and constructs the Document Object Model (DOM).

What is the DOM?

The DOM is an object representation of the HTML elements. It acts as a bridge between your code and the user interface, and has a tree-like structure where each node represents an element (such as a tag) in the document.

5. Output


DOM
HTML
BODY
DIV
H5
H1
UL
LI
LI
LI
UI
acme.com/engineers
Meet the
Engineers
A. Lovelace
G. Hopper
M. Hamilton

5. Output

You can manipulate the interfaces, by using DOM methods.

- HTML DOM **methods** are actions you can perform (on HTML Elements).
- HTML DOM **properties** are values (of HTML Elements) that you can set or change.

```
<h2>My First Page</h2>

<p id="demo"></p>

<script>
     document.getElementById("demo").innerHTML = "Hello World!";
</script>
```

6. JavaScript Core

The next lessons, you will learn the core concept of JavaScript:

- Objects
- Arrays and array methods
- Functions and Arrow Functions
- Destructuring
- Template literals
- Ternary Operators

Thank you

Perfect Practice, Does Perfect Thing