



# INTRODUCTION TO JAVASCRIPT

FOR THE WEB



HTML

Define the structure



CSS

Make it look nice



JavaScript

Make switches work

# CONTENT

---

- What is JavaScript?
- The developer console (`console`)
- Important programming concepts
  - Variables (`var`)
  - Control Structures (`if`)
  - Functions and events (`function`)
  - Loops (`for` / `while`)
- DOM manipulation
  - Add elements
  - Modify elements
  - Delete elements
  - Delete all child elements
- Add / remove CSS classes

## WAS IST JAVASCRIPT?

---

JavaScript is a scripting or **programming language** that allows you to implement complex things on web pages – every time a web page does more than just sit there and display static information for you to look at – displaying timely **content updates, interactive** maps, **animated** 2D/3D graphics [...] – you can bet that JavaScript is probably involved.

It is the **third layer** of the layer cake of standard web technologies (HTML / CSS / JS).

Source: [MDN](#)

# THE DEVELOPER CONSOLE

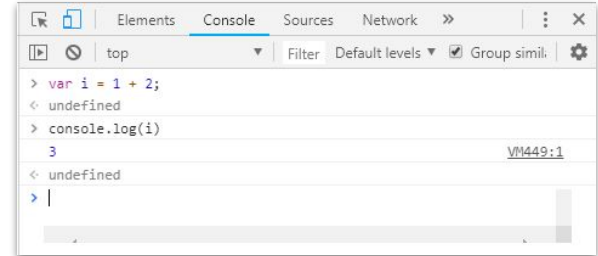
---

`console` is an **object** representing the browser's console

`console.log("Hi there!")`

`log` is a function that this object provides

F12 opens the **console** in Chrome



**Variables** are a named containers for values.

The keyword **var** defines a variable

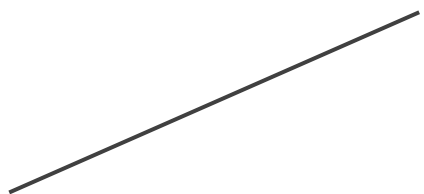
```
var q = 2;  
var price = 3.99;  
var total = q * price;  
var name = "Chris";  
var heading = document.querySelector("#h1");
```

The keyword **if** defines a **conditional code block**. It is only executed when the condition evaluates to true.

```
if(age >= 18) {  
    return true;  
} else {  
    return false;  
}
```

The **else** part is optional and is executed when the condition of the **if**-statement evaluates to **false**.

The keyword `function` defines a **function**, which is a reusable block of code with a name.



```
function updateName() {  
    var name = prompt("Enter new name");  
    para.textContent = "Player 1: " + name;  
}
```



We can add **event listeners** to any element. The click event listener allows us to react to an element being clicked on.

```
para.addEventListener("click", updateName);
```

An event listener needs to know what to do in case of the click event. We define that by passing a function name.

With **Javascript** and **functions**, we can make a website interactive. For example, we can define an action for a button our website:

```
var button = document.querySelector('#btn');  
button.addEventListener('click', convert);
```

```
function convert() {  
    ...  
}
```

The keyword **for** introduces a loop.  
Everything within the brackets is executed  
for each iteration of the loop.

A **for** loop is useful when you know how often you want to iterate the code block.

```
for(var i = 0; i < 10; i++) {  
    console.log(i);  
}
```

The keyword **do** introduces a loop.  
Everything within the brackets is  
executed for each iteration of the loop.

```
do {  
    ...  
} while ( 1 == 1)
```

A **while loop** repeats a block  
of code for as long as a  
condition is true.

// 1. Get the element you want to change

```
var task = document.querySelector("#task1");
```

// 2. Change the property you want

```
task.innerHTML = "<b>Do homework</b>";
```

// 1. Get the parent element of the new element

```
var taskList = document.querySelector("#tasks");
```

// 2. Create a new element

```
var newTask = document.createElement("li");
```

// 3. Define the element's content

```
newTask.textContent = "Learn CSS";
```

// 4. Add the new element to it's parent

```
taskList.appendChild(newTask);
```

# DOM MANIPULATION

## DELETE ELEMENTS

---

// 1. Get the element you want to remove

```
var task = document.querySelector("#task1");
```

// 2. Get the parent node of that element

```
var parent = task.parentNode;
```

// 3. Remove the task from the parent

```
parent.removeChild(task);
```

## DOM MANIPULATION

### DELETE ALL CHILD ELEMENTS

---

```
// 1. Get the element you want clear
```

```
var taskList = document.querySelector("#tasks");
```

```
// 2. Loop through children and remove each one
```

```
while (taskList.firstChild) {  
    taskList.removeChild(taskList.firstChild);  
}
```



## CSS CLASSES

ADD / REMOVE

---

```
// Get the element you want to ass a CSS class to
```

```
var taskList = document.querySelector("#tasks");
```

```
// Add a CSS class
```

```
taskList.classList.add("done");
```

```
// Remove a CSS class
```

```
taskList.classList.add("open");
```