

FUNCTIONS

Functions

A function is a way of writing your code so that you can reuse it instead of writing the same lines of code everywhere where the same thing needs to be done. A function also makes it much easier to overview your code.

```
myFunction()
```

Syntax

A function starts with **function** followed by the **function name**, which you decide entirely yourself. Then comes a set of **parentheses** followed by a set of **curly braces**, and inside these the **code** that the function will execute.

```
function myFunction()  
{  
    // Code here  
}
```

VARIABLE & CONSTANTS

Variable & Constants

A variable is a reference to a value. The three primary variable types are **var**, **let**, and **const**. The **const** variable's value cannot be overwritten once it is declared. **Var** and **let** can, but it is recommended to use **let** for variables, which value is to be overwritten.

const, let, var

Syntax

```
var myVariable = value
let myVariable = value
const myVariable = value
```

Variables are defined by first writing their type (**const**, **let** (or **var**)). This is followed by the name of the variable, which acts as the variable's reference.

After this comes an **equals sign** followed by the value of the variable.

SELECTORS

Selectors

A selector is used to select a specific element in the HTML. The three primary ones are **class**, **id**, and the element itself. Below is how to access each of them.

class, **id**, **element**

Syntax

Class: **.class-name**
Id: **#id_name**
Element: **element-name**

To access an element via its **class** in CSS, a **period** is used followed by the **class name**. If an element is to be accessed by its **id**, it is a **hashtag** followed by the **id's name**. If you access it by the element itself, nothing comes before it.

EVENTLISTENERS

Eventlisteners

An eventlistener is a way of listening to events. All interactions with the browser are events (keystrokes, clicks, inputs, etc.). They are used to run some **code** when the specified event is fired.

```
.addEventListener()
```

```
variableName.addEventListener  
('eventtype', funktionsNavn)
```

Syntax

Eventlisteners are added using **.addEventListener()**. This is appended to the **variable**. The event is in the quotation marks inside the **parentheses**. After this comes a comma followed by the **name of the function** to be run when the event is triggered.

CLASS MANIPULATION

Class manipulation

Class manipulation is to add **classes** to and/or remove **classes** from an element. The three primary methods that you must know are `.add()`, `.remove()` og `.toggle()`.

```
.add(), .remove(), .toggle()
```

```
variable.classList.add('class-name')  
variable.classList.remove('class-name')  
variable.classList.toggle('class-name')
```

Syntax

The syntax is similar to that of an eventlistener, which means **variable name** followed by `.classList`, which is again followed by **period** and **method name**. Inside the **parentheses** is a set of quotes with the **class** to be added or removed.

TIMERS

Timers

Timer functions are used to cause something to happen either *after* a given interval has passed or *every time* a given interval has passed. The most used are `setTimeout` and `setInterval`. `setInterval` runs *every time* the specified interval has passed, while `setTimeout` only runs *once* after the specified interval has passed.

`setInterval()/setTimeout()`

```
// setInterval
setInterval(function() {
    // Code here
}, interval)

// setTimeout
setTimeout(function() {
    // Code here
}, interval)
```

Syntax

The syntax is the same for `setInterval` and `setTimeout`. They start with their name followed by `parentheses`, inside which is `function` followed by `parentheses` and `curly brackets`. In those is the `code` to be run. The interval is as shown on the left.

RANDOMNESS

Randomness

Randomness is used when something needs to be generated randomly. In programming, it is used when you want to let the computer take over some of the control of a certain action. In JavaScript, `Math.random()` is used to generate a random value (by default between 0 and 1).

0-1

Syntax

You start by writing `Math`, which is then followed by `.random` and a set of `parentheses`, so that it becomes `Math.random()`, which gives a number between 0 (included) and 1 (excluded). See the examples on the left.

```
// Random decimal number between 0 and 1
Math.random()

// Random decimal number between 1 and 10
Math.random() * 10

// Random integer between 1 and 10
Math.floor(Math.random() * 10) + 1
```

OPERATORS

Operators

Used to determine a value from two or more numbers. In programming, they are used to check if something is true (for example, if one number is equal to another).

&&, ||, >, <
<=, >=, =, ==

&&	(and)
	(or)
>	(greater than)
<	(less end)
<=	(less than or equal to)
>=	(greater than or equal to)
=	(equals)
==	(equals the same value)

Syntax

There is no syntax as such for the **operators** alone, as these are just symbols that perform a certain action, and are therefore always used in conjunction with something else (**if** statements).

CONDITIONS

Conditions

Conditions are used to check whether a statement is true or false. Imagine that you only have room for ten items in your basket. For each item you put in your basket, you check whether it is full. This is false until the tenth item is placed and the basket is thus full.

if/else

Syntax

```
if(condition)
{
    // Code here
}
```

An **if** statement starts with **if** followed by **parentheses** containing the **condition**. After this come the **curly brackets**, where the **code** to be run is.

CONSOLE

Console

The console is the primary troubleshooting tool. The error messages are given here, and it's also the place to check the code for bugs. The hotkeys for opening the console in Chrome and Firefox on Mac and Windows are shown on the right.

	MacBook:
Chrome:	⌘ + Option + J
Firefox:	⌘ + Option + K
	Windows:
Chrome:	CTRL + Shift + K
Firefox:	CTRL + Shift + K

```
console.log()
```

Syntax

There is no syntax for the console as plain JavaScript is written in it. To log to the console, type **console** followed by **.log**. Then **parentheses** containing what you want to log.

AUDIO

Audio

To play audio in the browser with JavaScript, you must have an audio element with a **src attribute** in your HTML and bring it in as a **variable** in your JavaScript. Then you can work with the audio with methods like `.play()`, `.pause()`, `.muted`, `.volume`, `.duration` and `.currentTime`.

Play audio	<code>.play()</code>
Pause audio	<code>.pause()</code>
Mute audio	<code>.muted</code>
Volume	<code>.volume</code>
Audio duration	<code>.duration</code>
Time in audio	<code>.currentTime</code>

```
// Element
<audio src=""></audio>

// Variabel
const audio =
document.querySelector('audio')

// Play audio
audio.play()
```

Syntax

You must have an audio element in your HTML in order to work with audio, which is brought in as a **variable** in the JavaScript. On the left is shown how to play sound.

EXAMPLES

Functions

Variables

Selectors

Events

Class manipulation

Timers

Randomness

Operators

Conditions

Console

Audio