



Javascript Beginner Tutorial

Cheatsheet

By: supersimple.dev

Tutorial link: <https://www.youtube.com/watch?v=bArwRwHey6c>

Intro

Programming = giving instructions to a computer.

The computer will follow your instructions exactly.

- Here are the Javascript instructions we learned in this course
- Use these instructions and combine them into more complex instructions
- Eventually, the complex instructions come together to form a full software application

```
alert('hello');
```

Opens a popup with the text "hello" inside

```
console.log('hi');
```

Displays the text "hi" in the Chrome console

```
document.body.innerHTML = 'hello';
```

Replaces entire web page with the text "hello"

```
<button>Click</button>
```

Creates a button with the text "Click" inside

```
<button>Click</button>
```



opening tag



closing tag

```
<script>
```

```
  alert('hello');
```

```
</script>
```

Runs the Javascript code inside the `<script>` tags

```
'hello'
```

```
"hello"
```

String = a value in Javascript that represents a piece of text

Strings can be written like `'...'` or `"..."`

<pre>let myVar;</pre>	Variable = a container that can store a value
<pre>let name = 'Simon';</pre>	Creates a variable named <code>name</code> that stores the string <code>'Simon'</code>
<pre>let age = 27;</pre>	Creates a variable named <code>age</code> that stores the number 27

<pre>console.log(name);</pre>	Access the value inside a variable by typing the variable name
<pre>console.log('name');</pre>	Will <code>console.log('Simon')</code> ; since <code>name</code> is a variable
	Will <code>console.log('name')</code> ; since <code>'name'</code> is just a string

<pre>typeof 'hello'</pre>	Returns the type of the value: <code>'string'</code>
<pre>typeof 27</pre>	<code>'number'</code>
<pre>typeof name</pre>	Returns the type of the value inside the variable: <code>'string'</code>

<pre>'Hello ' + 'World'</pre>	Combine 2 strings together: <code>'Hello World'</code>
<pre>4 + 5 * 3</pre>	Do math on numbers: 19
<pre>(4 + 5) * 3</pre>	27
<pre>(4 - 2) / 2</pre>	1

<pre><html></pre>	All code for the website should be inside <code><html></html></code>
<pre> <head></pre>	<code><head></code> contains information about the page
<pre> <title>Title</title></pre>	<code><title></code> sets the title in the tabs
<pre> </head></pre>	
<pre> <body></pre>	<code><body></code> contains everything that appears <u>on</u> the page
<pre> <button>Click</button></pre>	
<pre> ...</pre>	
<pre> </body></pre>	
<pre></html></pre>	

<pre><div>Text</div></pre>	Creates an invisible box on the page, containing whatever is inside the <code><div></code> tags (can be text or other HTML elements)
--	--

<pre><div></pre>	This <code><div></code> displays an invisible box containing 2 buttons
<pre> <button>Button1</button></pre>	
<pre> <button>Button2</button></pre>	
<pre></div></pre>	

<pre>let div = document.createElement('div');</pre>	Creates an HTML <code><div></code> element and saves it in a variable
---	---

```
div.innerText = 'Hello';
document.body.appendChild(div);
```

Sets the text inside the div (<div>Hello</div>)
Adds the <div> to the end of <body>

```
function addTodo() {
  console.log('Add Todo');
}
addTodo();
```

Creates a function named `addTodo`

Runs the code inside the function

```
function addTodo(todoTitle) {
  console.log(todoTitle);
}
addTodo('Wash car');
```

Create a parameter named `todoTitle` (parameter = variable that can be used inside the function)

Set the value of the parameter to `'Wash car'`

```
function addTodo(todoTitle, dueDate) {
  // ...
}
```

function name **parameter(s)**

↑ ↑

function body

↙

Function Terminology:

1. Creating a function = "defining" a function
 2. Running a function = "calling" a function
 3. `function addTodo(todoTitle) {}` = the function "takes" a parameter named `todoTitle`
 4. `addTodo('Wash car');` = we "pass in" an "argument" (a value `'Wash car'`) to the function
-

```
let todos = [
  'Get groceries',
  'Wash car'
];
```

Array = a list of values
Creates an array of string values and saves it in a variable named `todos`

```
todos.push('new todo');
todos.pop();
```

Adds new value to the end of the array
Removes the last value from the array

```
todos.push('new todo');
```

Method = function attached to a value



method

```
'hello'.toUpperCase();
```

Gives the uppercase version of the string

```
todos.forEach(function (title) {  
  // ...  
});
```

Loop = for each value in the array, it will:
1. Save the value in the `title` parameter
2. Run the inner function

```
<button disabled="true">Click</button>  
<button hidden="true">Click</button>
```

HTML attribute = modifies the appearance and behaviour of HTML elements

```
<button onclick="console.log('hi')">  
  Click  
</button>
```

onclick attribute = runs the code inside the quotes "... " when button is clicked

```
<input type="text" />  
<input type="checkbox" />  
<input type="date" />
```

Creates a text box
Creates a checkbox
Creates a date picker

```
<input id="text-box" type="text" />
```

id attribute = allows you to get the element and manipulate it using JavaScript

```
let textbox = document.getElementById('text-box');
```

Gets the HTML element using its `id` and saves it in a variable

```
let value = textbox.value;
```

Gets the text that's currently inside the textbox and saves it in a variable

```
// comment  
/*  
  multi line  
  comment  
*/
```

Add comments to add extra info. They will be ignored by the computer

```
<div id="my-div"></div>
<script>
  let div = document.getElementById('my-div');
  let button = document.createElement('button');

  div.appendChild(button);           Add element inside <div id="my-div"></div>
  div.innerHTML = 'hello';          Erases everything inside <div id="my-div"></div>
                                     and replaces it with what's inside the string ('hello')
  div.innerHTML = '';               To erase everything in the div, just use empty string ''
</script>
```


```
let num = 5;                          Creates a variable that can be reassigned
num = 6;
let name;                              Create a variable without assigning it a value

const num = 5;                         Creates a variable that can not be reassigned (constant)

var num = 5;                           Old way of defining a variable (generally not used anymore)
```

```
let person = {
  name: 'Taylor',
  age: 27
};
```

property (key) **value**



```
person.name           Get the value of the name property: 'Taylor'
person.age            27
person['name']        Same as person.name write it like this if the property
                     contains hyphens: person['first-name']
```

```
const property = 'name';   Same as person['name']
person[property];
```

```
<button style="background-color: red;
font-size: 18px">Click</div>
```

Style attribute = changes the appearance of the HTML element using CSS

CSS syntax

```
style="background-color: red; font-size: 18px"
```



style name



style value

```
const btn = document.createElement('button');  
btn.style = 'background-color: red';  
btn.disabled = true;  
btn.hidden = true;
```

Set HTML attributes using JS. This is a feature of the Document Object Model (DOM)

```
const id = new Date().getTime();  
btn.id = id;  
btn.onclick = function () {};
```

Gets the current time in milliseconds
Set the button's id using JavaScript
Set the onclick attribute

```
const id = btn.id;
```

Get the value of an attribute

```
function onClick(event) {  
    event.target;  
}  
btn.onclick = onClick;
```

When the button is clicked, this function gets an `event` object containing information about the click event
Gets the HTML element that was clicked

```
function func() {  
    return 100;  
}
```

Sets the return value to `100`

```
const result = func();
```

Save return value in a variable

```
function func() {  
    return 'one hundred';  
}
```

Sets the return value to a string

```
true  
false
```

Boolean = a JavaScript value representing whether something is true or false

```
typeof true
```

```
'boolean'
```

```
1 < 5  
1 > 5  
1 === 5  
1 !== 5  
1 >= 5  
1 <= 5
```

Comparison operators = compares 2 values and returns whether the result is `true` or `false`

```
let result = 1 < 5;
```

boolean values can be saved in a variable

```
'apple' > 'banana'
```

Alphabetical comparison. Returns `false`

```
if (result) {  
  console.log('If 1');  
} else if (result2) {  
  console.log('If 2');  
} else {  
  console.log('Else');  
}
```

Runs this line if `result` is `true`

Otherwise, runs this line if `result2` is `true`

Runs this line if all conditions are `false`

```
let array = [1, 3, 5, 7, 9];  
array.filter(function (num) {  
  if (num > 5) {  
    return true;  
  } else {  
    return false;  
  }  
});
```

`filter` = creates a new copy of the array and loops through each value. If the inner function returns `true` keep the value in the new copy. If the inner function returns `false` remove the value from the new copy.

Does not modify the original array.

```
+'99'  
'' + 99
```

Convert a string to a number

Convert a number to a string

MVC (Model View Controller)

We split our code into 3 sections:

1. Model = saves and manages data
2. View = manages how to render the data onto the web page
3. Controller = responds to interactions (user clicks a button), tell the model and view to update

More info: <https://en.wikipedia.org/wiki/Model-view-controller>

MVC Flow

1. View code renders the web page using the data in the Model
2. User clicks a button on the web page
3. Controller code responds to the click and then tells the Model to update its data
4. After the Model updates, the Controller tells View to re-render using the updated data

```
const personString = JSON.stringify({
  name: 'Taylor'
});
const personObject = JSON.parse(personString);

localStorage.setItem('key', 'data');
localStorage.getItem('key');
Array.isArray(myVar);
```

Converts an object into a string
'{"name": "Taylor"}'

Converts a stringified object back into an object

Saves a string in browser under the key: 'key'

Retrieves the string saved under 'key'

Checks if a variable is an array

```
const func = function () {
  console.log('hi');
};

const func = () => {};
const func = name => {
  console.log(name);
};
const func = (name, age) => {
  console.log(name);
  console.log(age);
};
const func = () => 'hi';

const createCounter = function () {
  let count = 0;
  return function () {
    count = count + 1;
    console.log(count);
  };
};
const count = createCounter();
count();
```

Alternative way of defining a function

Arrow functions
Brackets are optional if only 1 parameter

Brackets are required if 2 or more parameters

If all on one line, returns the value after =>

Usually when a function returns, all variables inside the function are deleted.

Closure = return a function from a function. The inner function will have permanent access to the variables in the outer function (count).

This will return a function. Calling the function will console.log(1);