# HOMEWORK

JavaScript From Beginner to Professional

NOVEMBER 20, 2022

CODE ACADEMY DOO
contact@codeacademy.mk

# CONTENTS

# INTRODUCTION

Our academy believes that homework is an essential part of all students' education. The purpose of completing tasks at home is to support students in becoming independent learners and encouraging a deeper understanding of the subject studied. It also develops skills in personal organization and a sense of responsibility. Home learning is an essential part of the preparation for examinations and it is vital that students dedicate time to learning key information required for exam success.

Research shows that there is the potential for adding 5 months to the progress of each student through their academy with the setting of effective homework tasks. Evidence also suggests that how homework relates to learning during normal school time is important. In order to be most effective, homework should be an integral part of learning, rather than an add-on. To maximize impact, it is important that students are provided with high-quality feedback on their work.

The optimum amount of homework is between 1-2 hours per class Common home learning activities may be reading or preparing for work to be done in class, or practicing and completing tasks or activities already taught or started in lessons, but it may include more extended activities to develop inquiry skills or more directed and focused work such as revision for exams. Where extended tasks are set over a number of weeks, these should be broken into a series of smaller steps for students to complete.

The purpose of homework is to keep you practicing the material we learn and what we are about to learn in the course. If you don't know how to solve any of the exercises, please first see the solution if it has one and try to figure it out, and if you still don't know how to solve them, contact the instructor.

The best way to learn is to practice. Happy coding!

# GRAMMAR AND TYPES

## Basics

JavaScript borrows most of its syntax from Java, C, and C++, but it has also been influenced by Awk, Perl, and Python.

JavaScript is case-sensitive and uses the Unicode character set. For example, the word Früh (which means "early" in German) could be used as a variable name.

```
const Früh = "foobar";
```

But, the variable früh is not the same as Früh because JavaScript is case sensitive.

In JavaScript, instructions are called statements and are separated by semicolons (;).

A semicolon is not necessary after a statement if it is written on its own line. But if more than one statement on a line is desired, then they must be separated by semicolons.

It is considered best practice, however, to always write a semicolon after a statement, even when it is not strictly needed. This practice reduces the chances of bugs getting into the code.

The source text of JavaScript script gets scanned from left to right, and is converted into a sequence of input elements which are tokens, control characters, line terminators, comments, or whitespace. (Spaces, tabs, and newline characters are considered whitespace.)

## Declarations

JavaScript has three kinds of variable declarations.

### var

- Declares a variable, optionally initializing it to a value.

### let

- Declares a block-scoped, local variable, optionally initializing it to a value.

### const

- Declares a block-scoped, read-only named constant.

## Variables

You use variables as symbolic names for values in your application. The names of variables, called identifiers, conform to certain rules.
A JavaScript identifier usually starts with a letter, underscore (_), or dollar sign ($). Subsequent characters can also be digits (0 – 9). Because JavaScript is case sensitive, letters include the characters A through Z (uppercase) as well as a through z (lowercase).
You can use most of ISO 8859-1 or Unicode letters such as å and ü in identifiers. (For more details, see this blog post or the lexical grammar reference.) You can also use the Unicode escape sequences as characters in identifiers. Some examples of legal names are Number_hits, temp99, $credit, and _name.

## Declaring variables

You can declare a variable in two ways:

- With the keyword var. For example, var x = 42. This syntax can be used to declare both local and global variables, depending on the execution context.
- With the keyword const or let. For example, let y = 13. This syntax can be used to declare a block-scope local variable. (See Variable scope below.)

You can declare variables to unpack values using the destructuring assignment syntax. For example, const { bar } = foo. This will create a variable named bar and assign to it the value corresponding to the key of the same name from our object foo.
Variables should always be declared before they are used. JavaScript used to allow assigning to undeclared variables, which creates an undeclared global variable. This is an error in strict mode and should be avoided altogether.

## Declaration and initialization

In a statement like let x = 42, the let x part is called a declaration, and the = 42 part is called an initializer. The declaration allows the variable to be accessed later in code without throwing a ReferenceError, while the initializer assigns a value to the variable. In var and let declarations, the initializer is optional. If a variable is declared without an initializer, it is assigned the value undefined.

```
let x;
console.log(x); // logs "undefined"
```

const declarations always need an initializer, because they forbid any kind of assignment after declaration, and implicitly initializing it with undefined is likely a programmer mistake.

```
const x; // SyntaxError: Missing initializer in const declaration
```

# DATA STRUCTURES AND TYPES

## Data types

The latest ECMAScript standard defines eight data types:

Seven data types that are primitives:

1. Boolean. true and false.
2. null. A special keyword denoting a null value. (Because JavaScript is case-sensitive, null is not the same as Null, NULL, or any other variant.)
3. undefined. A top-level property whose value is not defined.
4. Number. An integer or floating point number. For example: 42 or 3.14159.
5. BigInt. An integer with arbitrary precision. For example: 9007199254740992n.
6. String. A sequence of characters that represent a text value. For example: "Howdy".
7. Symbol. A data type whose instances are unique and immutable.

and Object

Although these data types are relatively few, they enable you to perform useful operations with your applications. Functions are the other fundamental elements of the language. While functions are technically a kind of object, you can think of objects as named containers for values, and functions as procedures that your script can perform.

## Data type conversion

JavaScript is a dynamically typed language. This means you don't have to specify the data type of a variable when you declare it. It also means that data types are automatically converted as-needed during script execution.

So, for example, you could define a variable as follows:

```
let answer = 42;
```

And later, you could assign the same variable a string value, for example:

```
answer = 'Thanks for all the fish!';
```

Because JavaScript is dynamically typed, this assignment does not cause an error message.

# EXERCISES

## Exercise 1.1

Declare a variable firstname and initialize it with the value 'Lata'.

## Exercise 1.2

Which value does x have after execution of the following code?

```
let x = 'Geeta';
```

## Exercise 1.3

Declare a variable flower and assign it the value 'rose'. Declare a second variable tree and assign it the value 'maple'.

## Exercise 1.4

Which value does x have after execution of the following code?

```
let x = 'Tic';
x = 'Tac';
x = 'Toe';
```

## Exercise 1.5

Which value does x have after execution of the following code?.

```
let x = 'Laurel';
let y = 'Hardy';
let z = y;
y = x;
x = z;
```

## Exercise 1.6

Evaluate each of the following Javascript expressions and show the value.

1. -9*3
2. “value is “ + 50
3. 17 % 5
4. 5 % 17
5. 5/10
6. (4 == 4)
7. (4 != 5)
8. (7 <= 8)
9. Math.ceil(x) - Math.floor(x)

## Exercise 1.7

Alert "Hello world."

## Exercise 1.8

Read a number (using prompt) and display it using alert.

## Exercise 1.9

Read two numbers and display their product like the example.
example: "The product of " + n1 + " and " + n2 + " is " + n1*n2

## Exercise 2.1

Read two numbers  (using prompt) and display their sum like the example.
example: "The product of " + n1 + " and " + n2 + " is " + n1 + n2

What problem did you encounter?

## Exercise 2.2

Read two numbers (using prompt) and display the larger like the example.
example: n1 + " is the larger."

## Exercise 2.3

Why pay a fortune teller when you can just program your fortune yourself?

- Store the following into variables: number of children, partner's name, geographic location, job title.
- Output your fortune to the screen like so: "You will be a X in Y, and married to Z with N kids."

## Exercise 2.4

Want to find out how old you'll be? Calculate it!

- Store your birth year in a variable.
- Store a future year in a variable.
- Calculate your 2 possible ages for that year based on the stored values.
- For example, if you were born in 1988, then in 2026 you'll be either 37 or 38, depending on what month it is in 2026.
- Output them to the screen like so: "I will be either NN or NN in YYYY", substituting the values.

## Exercise 2.5

Ever wonder how much a "lifetime supply" of your favorite snack is? Wonder no more!

- Store your current age into a variable.
- Store a maximum age into a variable.
- Store an estimated amount per day (as a number).
- Calculate how many you would eat total for the rest of your life.
- Output the result to the screen like so: "You will need NN to last you until the ripe old age of X".

## Exercise 2.6

Calculate properties of a circle, using the definitions here.

- Store a radius into a variable.
- Calculate the circumference based on the radius, and output "The circumference is NN".
- Calculate the area based on the radius, and output "The area is NN".

## Exercise 2.7

It's hot out! Let's make a converter based on the steps here.

- Store a celsius temperature into a variable.
- Convert it to fahrenheit and output "NN°C is NN°F".
- Now store a fahrenheit temperature into a variable.
- Convert it to celsius and output "NN°F is NN°C."