Introduction to JavaScript

Variables and Functions

Week 3

ReDI

Agenda

1. Homework Review
2. Introduction to JavaScript
3. Data types and variables
4. Functions

Homework Review

- **Read**

  Article on CSS

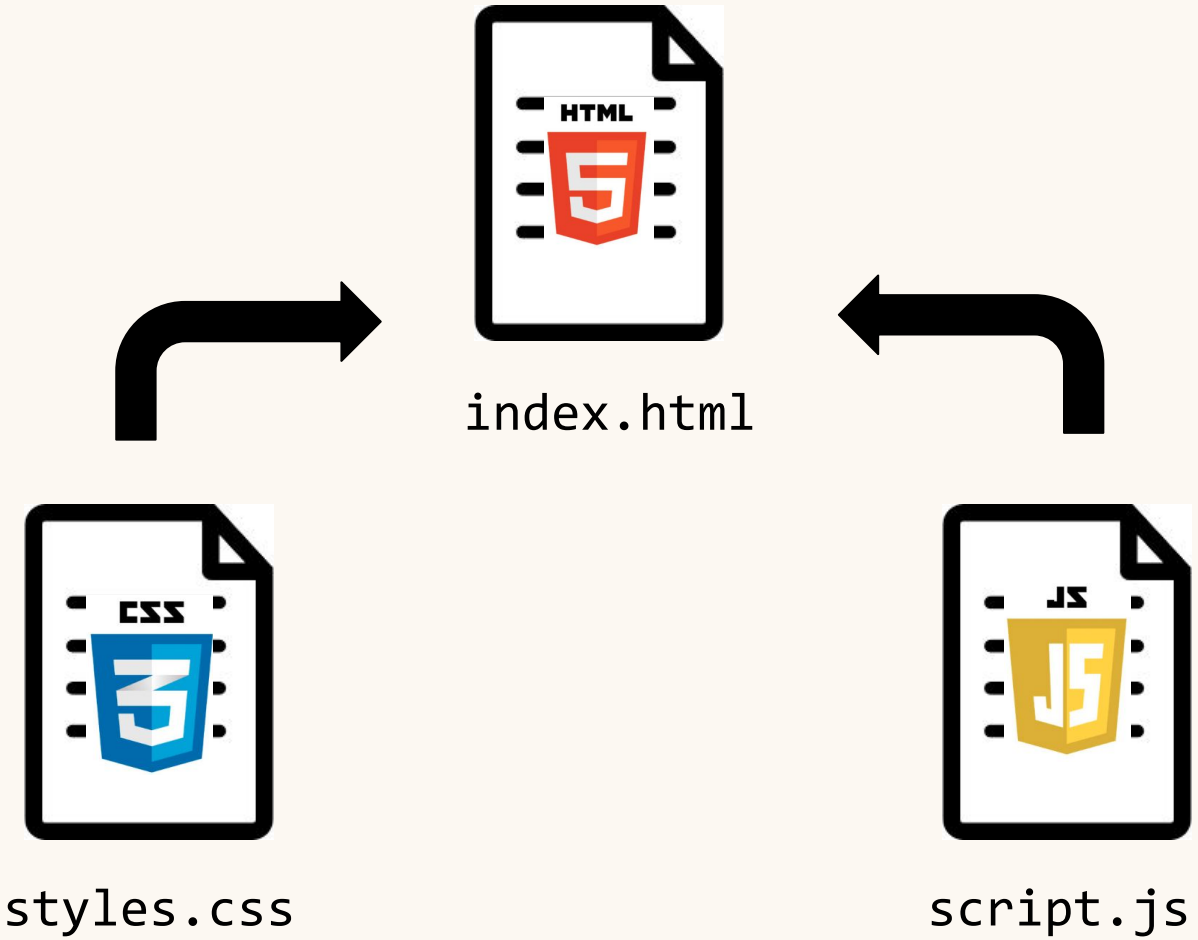
  A guide for future consultation about flexbox

- **Watch**

  Video about flexbox

  Introduction to JavaScript

The Perfect Setup



index.html


```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" type="text/css" href="styles.css">
    <title>JavaScript Funsies</title>
</head>
<body>
    <h1>JavaScript - Variables and Functions</h1>
</body>
<script src="script.js"></script>
</html>
```

Introduction to JS

JavaScript is a lightweight **programming language** that developers commonly use to create **more dynamic interactions** when developing web pages, applications, servers, and or even games.

Developers generally use JavaScript alongside HTML and CSS.

JavaScript manages **user interaction**, something that CSS cannot do by itself.

Browser

Javascript interacts with the browser.

What can you use in the browser?

- **Console**
- **Document**
- Screen
- Window
- Location
- ... and others

Console

The console is really helpful for developers to debug our code.

Debugging is the act of removing a **bug** in the code.

A bug is basically an **error** in the code.

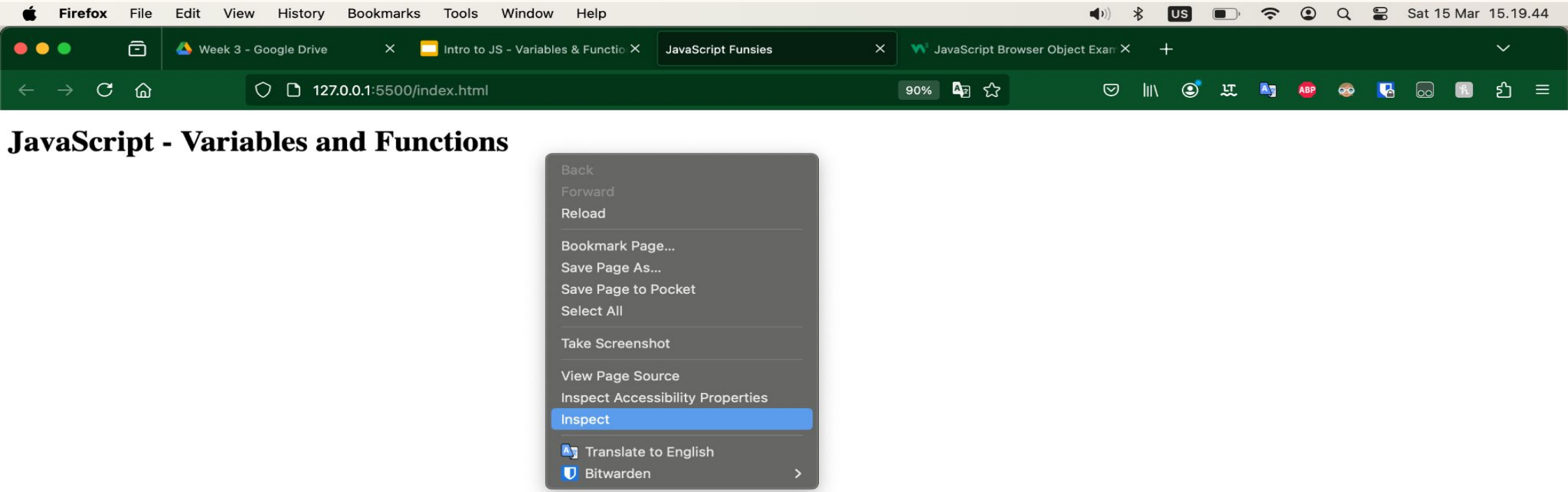
How can you use the console?

Console

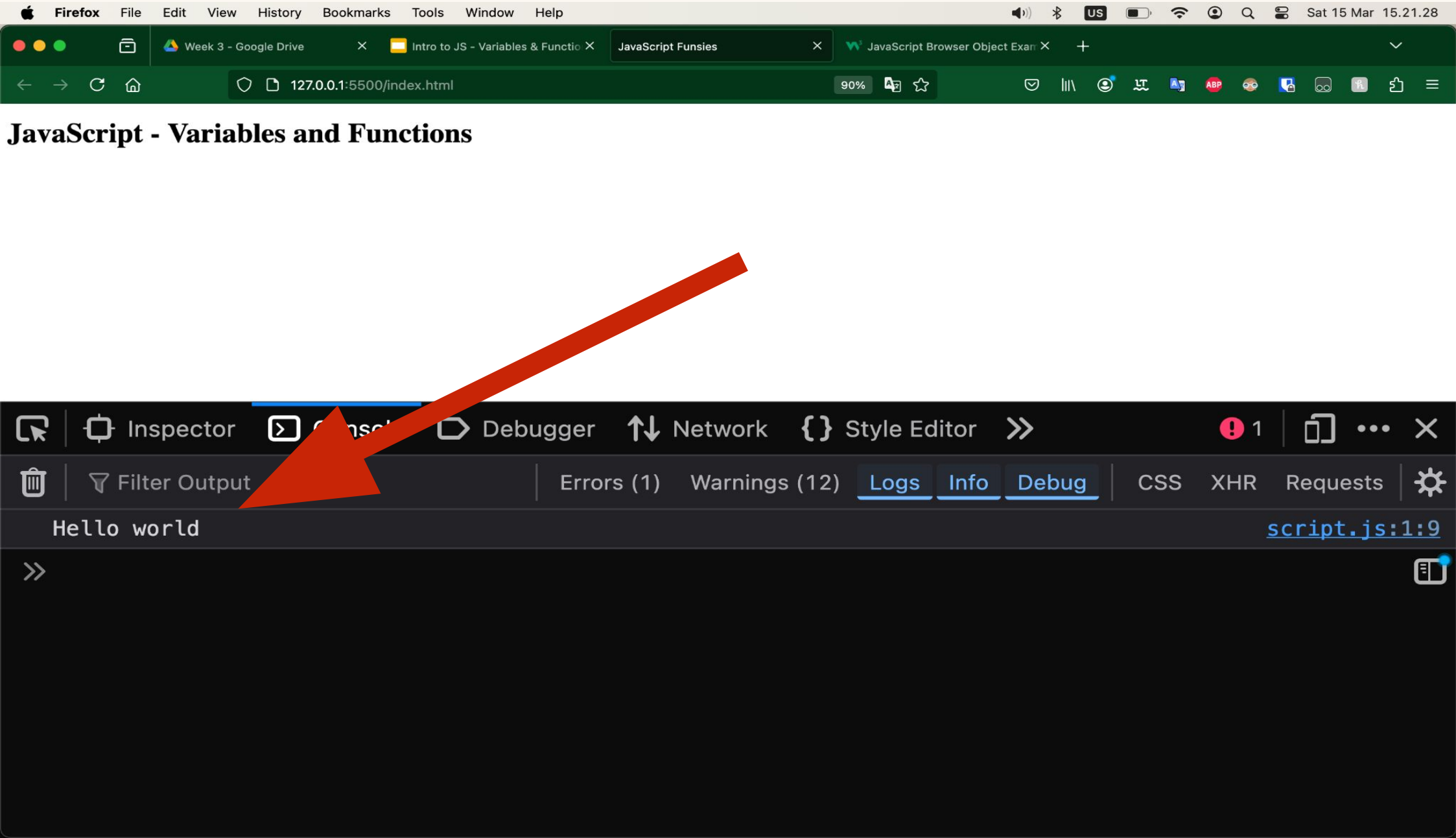
Open your script.js file in Visual Studio Code, type the following line and save the file:

```
console.log("Hello world");
```

Console

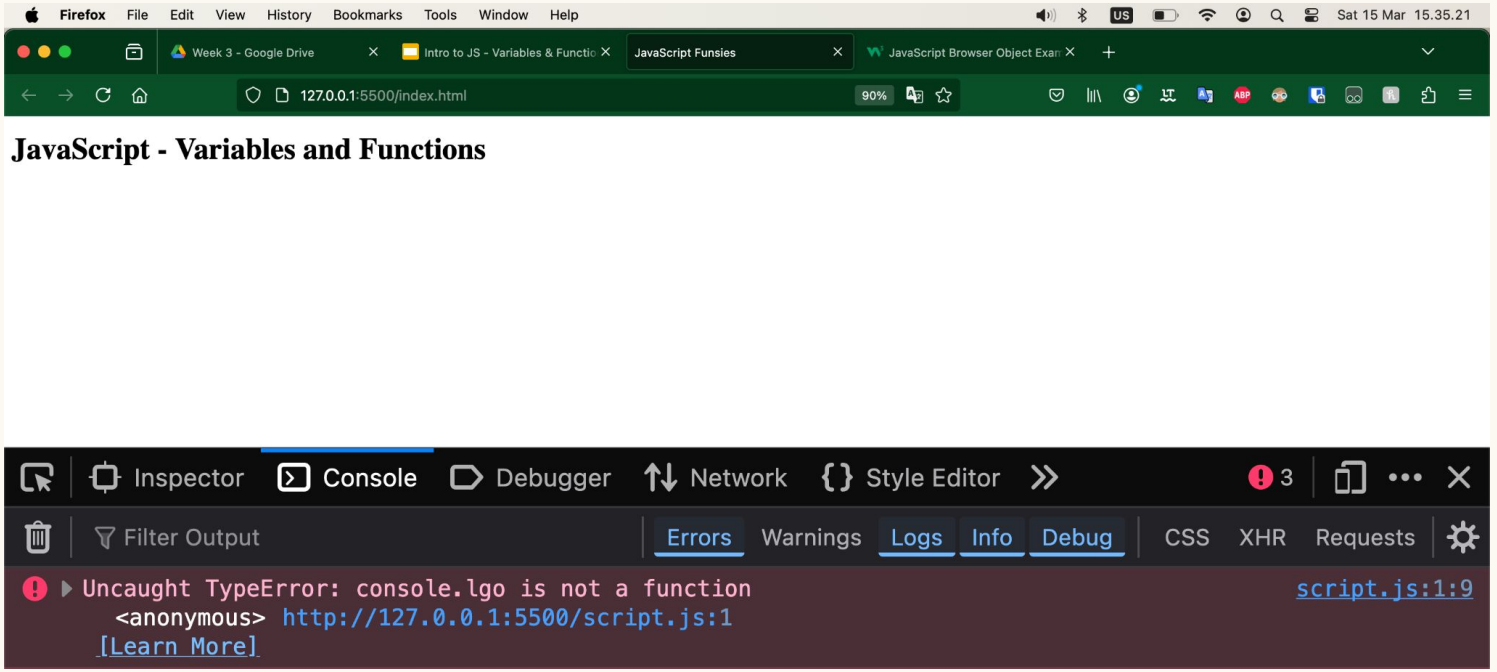

Firefox File Edit View History Bookmarks Tools Window Help
Sat 15 Mar 15.19.44
Week 3 - Google Drive
Intro to JS - Variables & Functio
JavaScript Funsies
JavaScript Browser Object Exan
127.0.0.1:5500/index.html
90%
JavaScript - Variables and Functions
Back
Forward
Reload
Bookmark Page...
Save Page As...
Save Page to Pocket
Select All
Take Screenshot
View Page Source
Inspect Accessibility Properties
Inspect
Translate to English
Bitwarden

Console


Firefox File Edit View History Bookmarks Tools Window Help
Sat 15 Mar 15.21.28
Week 3 - Google Drive
Intro to JS - Variables & Functio
JavaScript Funsies
JavaScript Browser Object Exam
127.0.0.1:5500/index.html
90%
JavaScript - Variables and Functions
Inspector
Debugger
Network
Style Editor
1
Filter Output
Errors (1)
Warnings (12)
Logs
Info
Debug
CSS
XHR
Requests
Hello world
script.js:1:9

Console

```
console.lgo("Hello world");
```



Document

Within a browser, JavaScript doesn't do anything by itself. You run JavaScript from inside your HTML web pages. To call JavaScript code from within HTML, you need the <script> element.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" type="text/css" href="styles.css">
    <title>JavaScript Funsies</title>
</head>
<body>
    <h1>JavaScript - Variables and Functions</h1>
</body>
<script src="script.js"></script>
</html>
```

Code Example


```
const name = "Lukas";

let age = 30;

age = 30 + 1;


console.log("Age: " + age);


//Accessing the HTML element <h1> with the id "heading"
const documentHeading = document.querySelector("#heading");


if (age > 30){
    documentHeading.textContent = "Old Lukas";
}
else {
    documentHeading.textContent = "Young Lukas";
}
```

JS - Basic Concepts

- Code is executed sequentially from the top to the bottom and left to right
- There are some reserved keywords that are part of the language such as: console, let, const, if, else ...
- Comments
  - // is used to comment out one single line
  - /* */ is used to comment out multiple sequential lines

Break

ReDI SCHOOL

Data types

Primitive values

- **Number** - Represents numeric values, inc. integers and decimals, e.g. `23`, `-5`, `1.05`, `0.005`
- **String** - Represents sequences of characters, enclosed in quotes (single or double), e.g. `“A text”`, `‘Another text’`, `“23”`, `‘0.005’`
- **Boolean** - Represents logical values - `true`/`false`
- **Undefined** - Represent the absence of a value
- **Null** - Represent exactly one value - `null`
- **Symbol** - Represents a unique and immutable identifier
- **BigInt** - Represent large integers, e.g. `12345678901234567890n`

Data types

Reference values (we will cover them later)

- **Object**

  ```
  {name: 'Lukas', age: 30}
  ```

- **Array**

  ```
  [1, 2, 3, 4]
  ```

- **Function**

  ```
  () => ({})
  ```

Variables

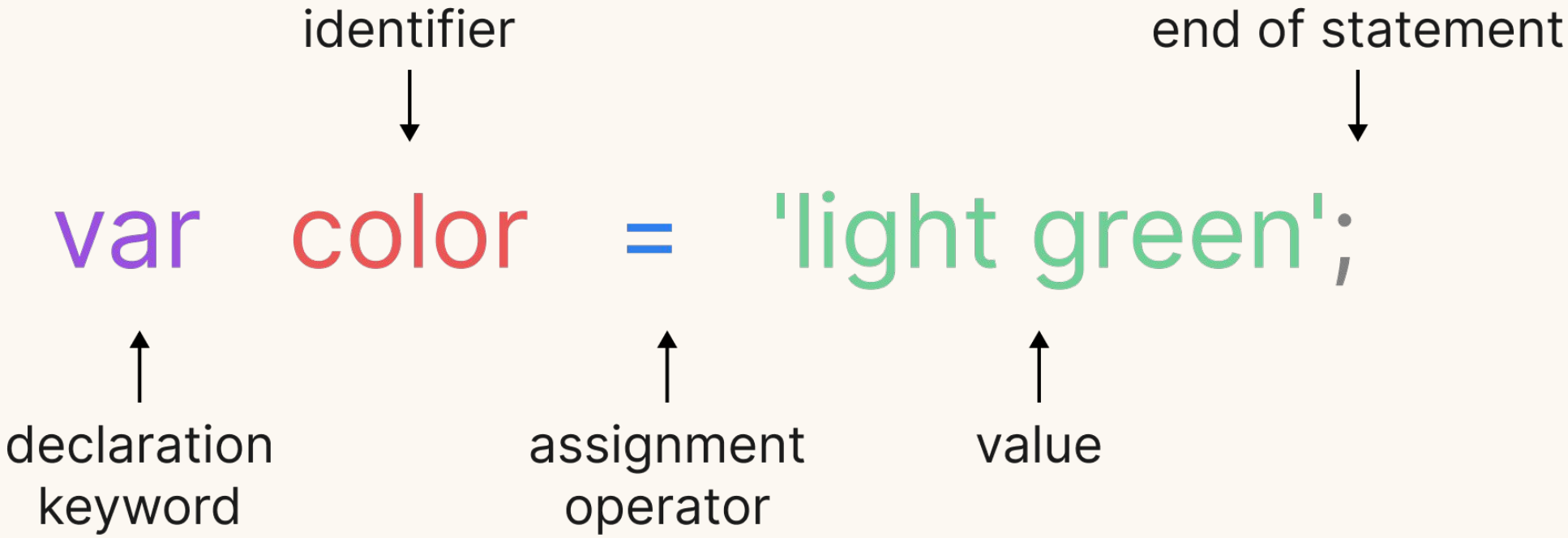
Variables are containers for storing **data**.

They “hold” the data value which can be changed anytime.

```
var myAge = 37;
var productPrice = 199;
var initialState = { };
```

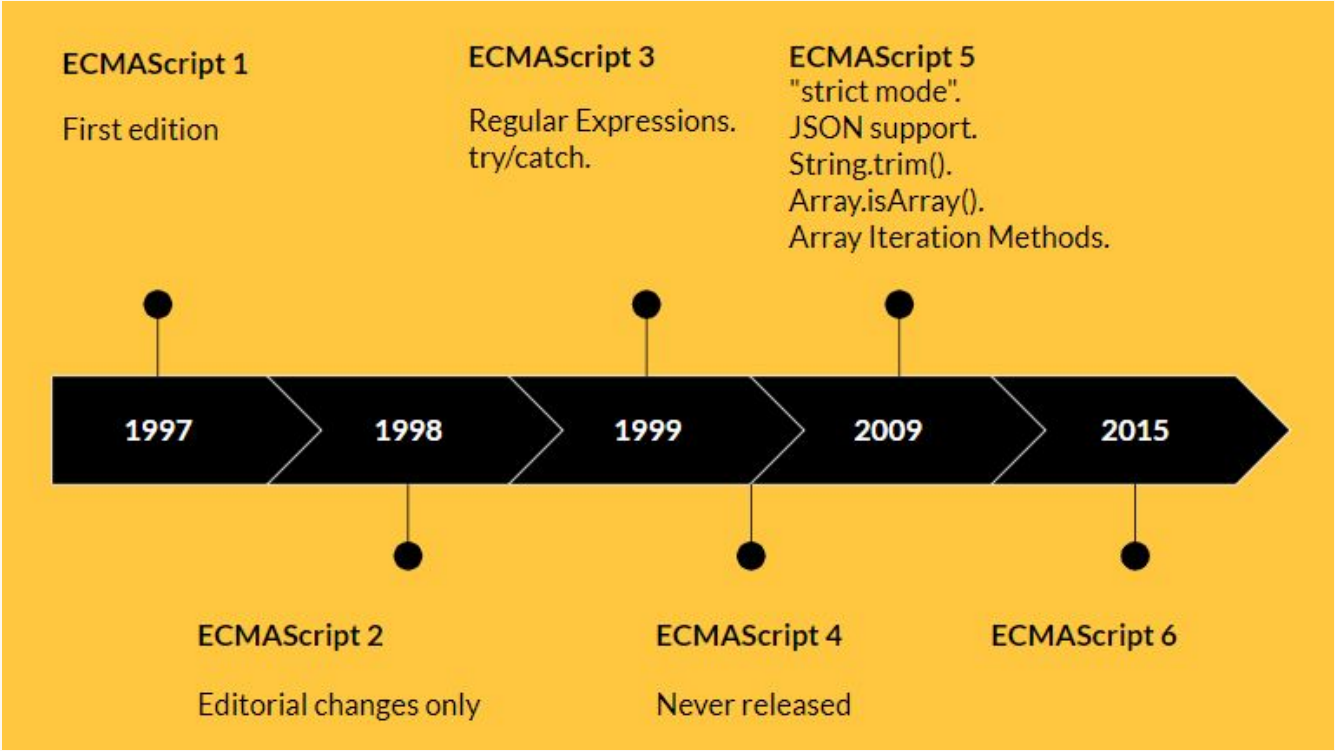
Variables

JavaScript uses **reserved keyword** **var** to declare a variable. A variable must have a unique name. You can assign a value to a variable using equal to (=) operator when you declare it or before using it.



ES5 vs ES6

ECMAScript is a standard specification of the language and the syntax. In ES6 some breaking new changes were introduced:



Variable declaration

Arrow functions

ES6 Variables

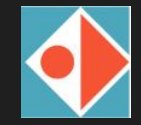
The reserved keyword `var` is not used anymore - instead we use:

- `let` - *can* be reassigned and modified
- `const`: is constant, so *cannot* be reassigned nor modified

ES6 Variables


```
const name = "Lukas";

let age = 30;
```



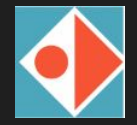
Declaration vs Initialization


```
let message;
message = "Hello World!";

let age = 30;
age = age + 1; //age = 30 + 1;

let address = 'Cylindervej';
address = 'Dieselvej';

const cprNumber = '123456-1234'; //Cannot change
```



Variable names

Variable names (or **identifiers**) have to be unique and can be short (like `x` and `y`) or have more descriptive names (`age`, `cprNumber`). If a variable name consists of more than one word we usually use `camelCase` (uppercase for the first letter of the next word: `cprNumber`).

Variable names - rules

1. Names can contain letters, digits and underscores.
2. Names must begin with a letter.
3. Names are case sensitive.
4. Reserved keywords cannot be used.
5. No spaces are allowed.

Exercise

1. Create 2 variables of type number/integer.
    - In a third variable try adding the numbers together.
    - Output the result.
2. Create 2 variables of type string (text).
    - Try creating a third variable that add them together.
    - Output the result.
3. Create 2 variables: a number and a string. Add them together. → What do you expect to happen? What happened?
4. Try re-initializing a const variable → What happened? Anything in the console? Try the same with let.
5. What is the difference between console.log(myName) and console.log('myName') when both would run without errors?

Type conversion

Variables can be changed from one **data type** to another by many means, such as **reassignment**, function and even **automatically** by Javascript.

```
3
4     let age = 29;
5     age = "Old"
6
```

Break

ReDI SCHOOL

Functions

When developing an application, you often need to perform the same action in many places.

For example, you may want to show a message whenever an error occurs.

To avoid repeating the same code at different places, you can use a function to wrap that code and reuse it.

Functions

There are 3 main steps to use functions:

1. Declare the function
2. Pass arguments(parameters)
3. Call the function

Functions

Declaring a function

To declare a function, you use the function keyword, followed by the function name, a list of arguments(if any) and the function body:

```
function sayHiToUser()
{
    alert("Hi");
}
```

Functions

Declaring a function

The function name must be a valid JavaScript identifier. By convention, the function names are in **camelCase** and start with verbs, e.g. **getData**(), **fetchContents**(), **isValid**(), etc.

A function can accept zero, one, or multiple arguments(parameters). In the case of multiple arguments, you need to use a comma to separate them.

Functions

Declaring a function

The following declares a function **sayHiToUser**() that takes no argument:

```
function sayHiToUser()
{
    alert("Hi");
}
```

Functions

Declaring a function

Inside the function body, you can write the code to implement an action. For example, the following sayHiToUser() function simply shows a message as an alert:

```
function sayHiToUser()
{
    alert("Hi");
}
```

Functions

Passing arguments

The following declares a function named **square** that takes one argument:

```
function square(number)
{
    number * number;
}
```

Functions

Passing arguments

And the following declares a function named **add** that takes two arguments:

```
function add(a, b) {
  a + b;
}
```

Functions

Calling a function!

To call a function, you use its name followed by arguments enclosing in parentheses like this:

```
sayHiToUser();
```

Functions

Calling a function!

If the function is expecting some arguments we need to pass them when we call it:

```
square(2); /* or */ add(2, 3);
```

Functions

Using a return value

One of the purposes of functions is to reuse the outcome of running some code.

Every function in JavaScript implicitly returns undefined unless you specify a return value

```
function square(number)
{
    return number * number;
}
```

Functions

Using a return value

```
function square(number)
{
    return number * number;
}


let squaredNumber = square(2);


console.log("Squared number: " + squaredNumber);
```

Exercise

20 minutes

A. Write a function named **calculateDogAge** that:

1. Takes 1 argument: your dog's age.
2. Calculates your dog's age based on the conversion rate of 1 human year to 7 dog years.
3. Outputs the result
4. Use the result from the function and create an alert with the human age of your puppy. Display the result to the screen like so: "Your doggie is XX years old in dog years!"

Exercise

20 minutes

B. Write a function named **moneyToSpendInCoffeUntil80** that:

    1. Takes 2 arguments: your current age and price for a cup of coffee

    2. Calculates how much money you will spend if you drink one cup of coffee per day until you are 80 years old.

Homework

You and your family are busy today and don't have time to cook for dinner. You've decided to order some food, but your partner and your children like different restaurants.

a. Write a function that takes 3 arguments: name of the restaurant, food and amount.
b. Outputs the result to your console with the message : You are ordering {amount} {food} from {name of the restaurant}
c. Call the function 3 times with different argument values