# Task

1. Create an account on GitHub

2. Check that you have Git in your machine

3. Configure Git

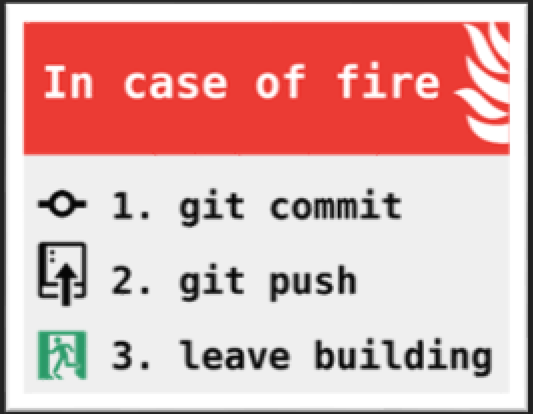
```
git config --global user.name "Your Name"

git config --global user.email "your.email@example.com"
```

## Github connection

Create a new repository in GitHub for the demo project

Connect your demo project and GitHub repository

- Open the terminal and locate demo project folder
- git init
- git add .
- git commit -m „first commit“
- git remote add origin URL
- git push origin master



In case of fire

1. git commit
2. git push
3. leave building

https://kbroman.org/github_tutorial/pages/init.html

## Programming JS

Learning JavaScript (or any programming language) is making mistakes and learning from them.

Do NOT get discouraged when you can't solve a problem or challenge.

## Data types

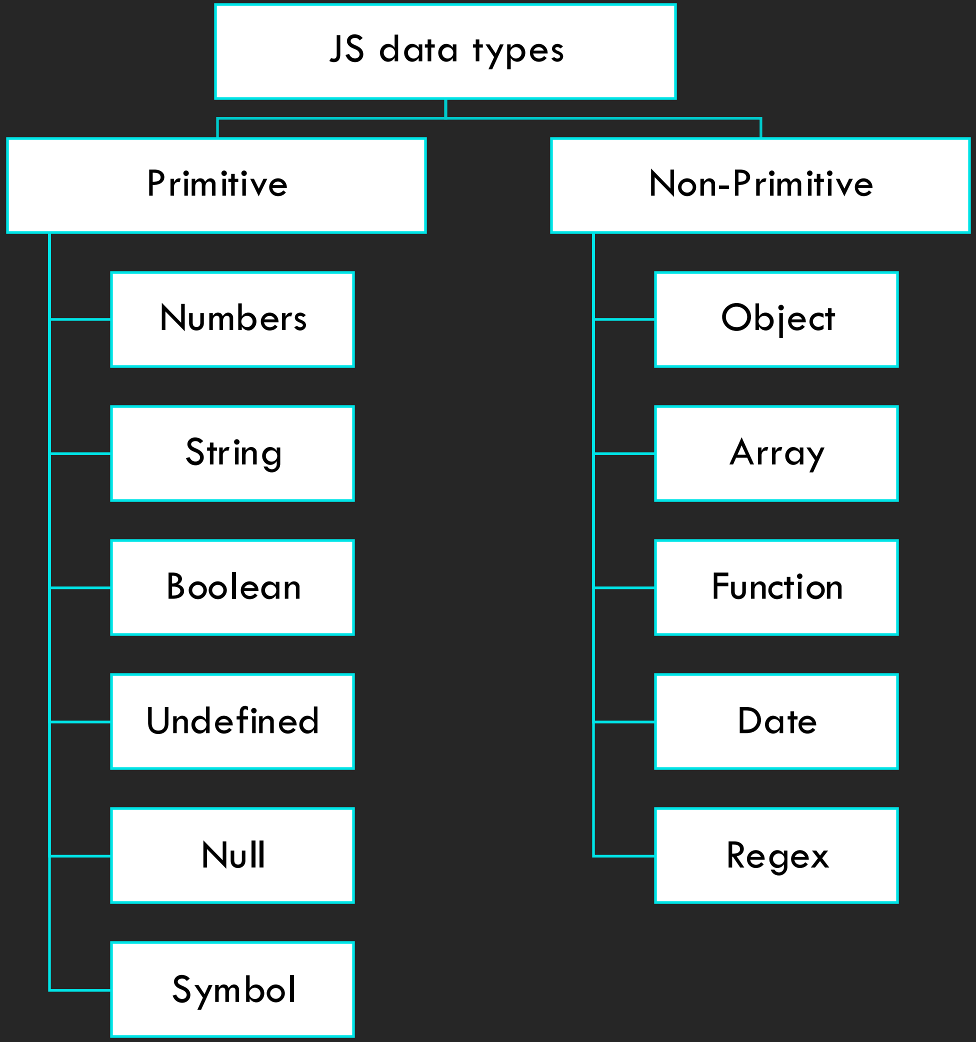
JavaScript is a dynamically typed language, meaning variables can hold values of any data type and the same variable can hold different types of values over its lifetime.

**Primitive**

- Number: Represents both integer and floating-point numbers. For example, 42 and 3.14.
- String: Used for textual data. It must be enclosed in quotes, e.g., "Hello, world!".
- Boolean: Represents a logical entity and can have only two values: true or false.
- Undefined: When a variable is declared but not assigned, it has the value undefined.
- Null: Denotes a null value, indicating the absence of any object value.
- Symbol (introduced in ES6): Used to create anonymous and unique values.

Complex

- Object: Collections of key-value pairs, where the values can be of any data type. Objects are used to store various keyed collections and more complex entities.



## Numbers

In JavaScript, these are all considered as Numbers.

Whether they are negative or positive, or even if they include decimal points, they fall under the category of numbers.

```
1
2
-5
3.5
2000
2021
-23.51
```

## Converting numbers

```
let answer = 42;
let stringAnswer = answer.toString(); // Converts to "42"
```

```
let str = '42';
let numericValue = Number.parseInt(str, 10); // Converts to 42
let numericValue = Number(str); // Can be used, but will not accurately convert a string that contains characters also. Eg: "42abc" will return NaN – Not a Number.
```

## Strings

JavaScript strings are a fundamental data type used for storing and manipulating text.

You can declare strings using either single quotes (' '), double quotes (" "), or backticks (``).

```
'This is a string’;
“This is another string!”;
`This is also a string!`;
```

## Boolean

A boolean is a data type that represents one of two values: **true** or **false**.

Booleans are commonly used in conditional statements to control the flow of a program.

## Boolean tests

|  |  |
|---|---|
| Greater than | 5 > 2 |
| Less than | 5 < 2 |
| Alphabetic | “apple” > “Apple” |

## Operators

## Arithmetic Operators

| Operator | Description |
|---|---|
| + | Addition |
| - | Subtraction |
| * | Multiplication |
| ** | Exponentiation (ES2016) |
| / | Division |
| % | Modulus (Remainder) |
| ++ | Increment |
| -- | Decrement |

## Operators - Assignment

| Operator | Example | Same As |
|---|---|---|
| = | x = y | x = y |
| += | x += y | x = x + y |
| -= | x -= y | x = x - y |
| *= | x *= y | x = x * y |
| /= | x /= y | x = x / y |
| %= | x %= y | x = x % y |
| **= | x **= y | x = x ** y |

## **Operators -** Comparison

| **Operator** | **Description** |
|---|---|
| == | equal to |
| === | equal value and equal type |
| != | not equal |
| !== | not equal value or not equal type |
| > | greater than |
| < | less than |
| >= | greater than or equal to |
| <= | less than or equal to |
| ? | ternary operator |

## **Operators -** Type

| **Operator** | **Description** |
|---|---|
| typeof | Returns the type of a variable |
| instanceof | Returns true if an object is an instance of an object type |

## Type Conversions

String()

value.toString()

Number()

+value

Boolean()

!!value

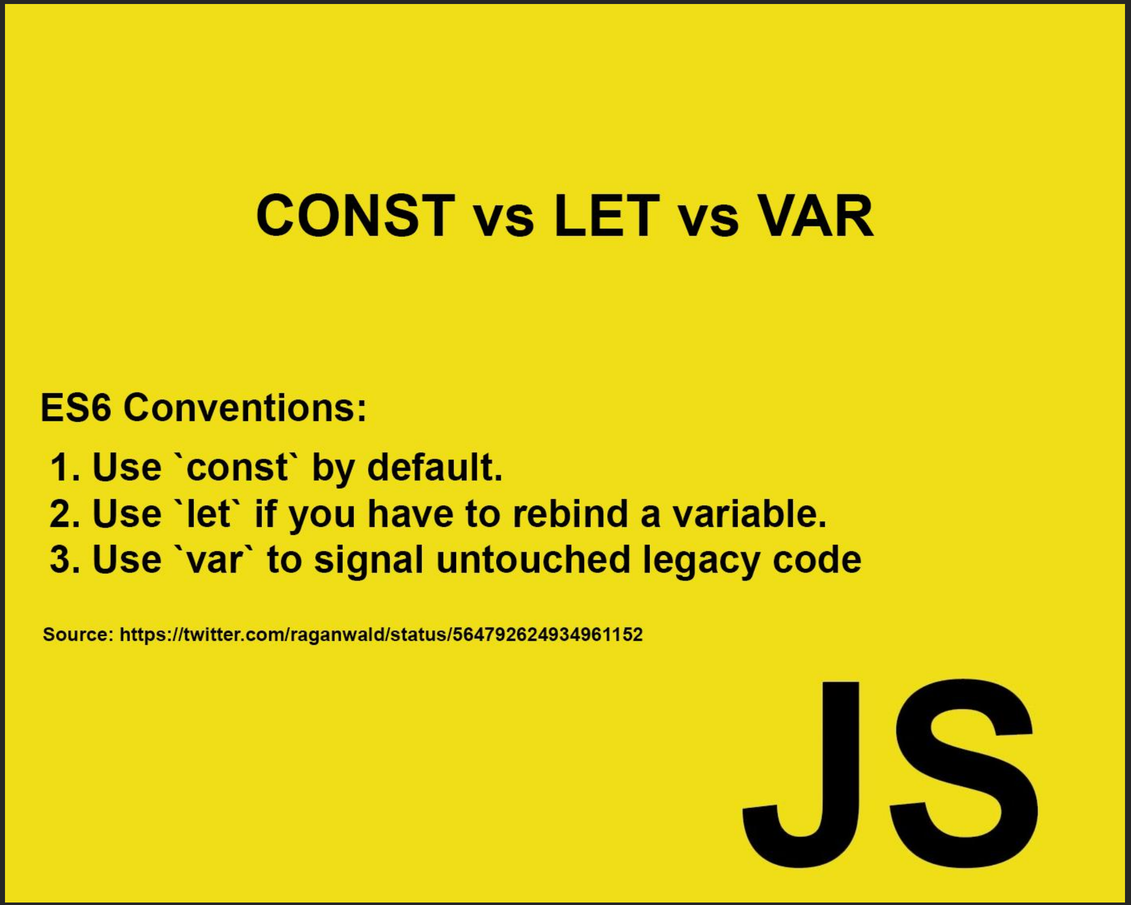
NOTE: While using shorthand operators like **+** and **!!** can make the code more concise, it's essential that their usage is clear to anyone reading the code. When used appropriately, these shorthands can make the code both clean and efficient.

## Variables

## Variables

To catch and hold values, JS provides a thing called **binding** or **variable**

- const
- let
- var



CONST vs LET vs VAR

ES6 Conventions:

1. Use `const` by default.
2. Use `let` if you have to rebind a variable.
3. Use `var` to signal untouched legacy code

Source: https://twitter.com/raganwald/status/564792624934961152

JS

## let and const

```
let name = 'Sam';
console.log(name);

let language = 'C++';
language = 'JavaScript’;

let sum = 0;
sum += 1;
```

```
const language = 'C++'; // Cannot be re-assigned anymore
console.log(language); // "C++"
language = 'Python'; // ❌ Type error: this will break your script
```

## JavaScript syntax

```
let x, y, z;          // Declare Variables

x = 5; y = 6;         // Assign Values

z = x + y;            // Compute Values
```

Note: JS is case-sensitive and hyphens are not allowed in JS

~~first-name~~ -> first_name OR FirstName OR firstName

# JS keywords

| | | | | | | |
|---|---|---|---|---|---|---|
| await | debugger | extends | import | package | static | void |
| break | default | false | in | private | this | while |
| case | delete | finally | instanceof | protected | throw | with |
| catch | do | for | interface | public | try | yield |
| class | else | function | let | return | True | |
| const | enum | if | new | super | typeof | |
| continue | export | implements | null | switch | var | |

## Task - variables

Declare two variables: student and name.

Assign the value ”Mikko" to name.

Copy the value from name to student.

Show the value of student using alert (must output “Mikko”).

```
let student, name; // can declare two variables at once

name = ”Mikko";

student = name;

alert( student ); // ”Mikko"
```

https://www.w3schools.com/js/js_exercises.asp

Variables, Operators, Data types

## Sum up

Data types – primitive // number, string, boolean, undefined, null, (symbol)

JS syntax // case sensitive, hypens not allowed, semicolon

Operators

- Arithmetic // + - * ** / % ++ --
- Assignment // = += -= *= /= %= **=
- Comparison // == === != !== < > <= >= ?
- Type // typeof()

prompt(), alert(), confirm()

## Tasks in GH

https://github.com/margittennosaar/JS_REACT25K