Front-End Web Development

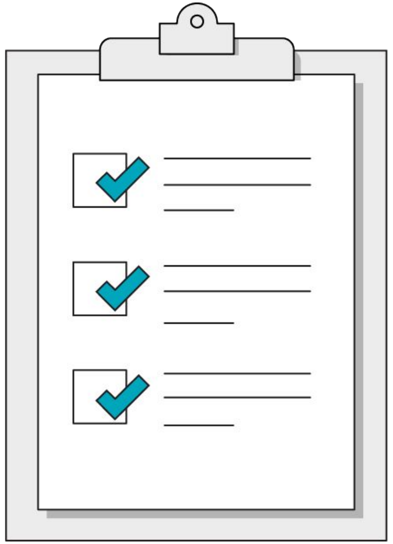
Introduction to Programming With JavaScript

GENERAL ASSEMBLY

Today’s Learning Objectives

In this lesson, you will:

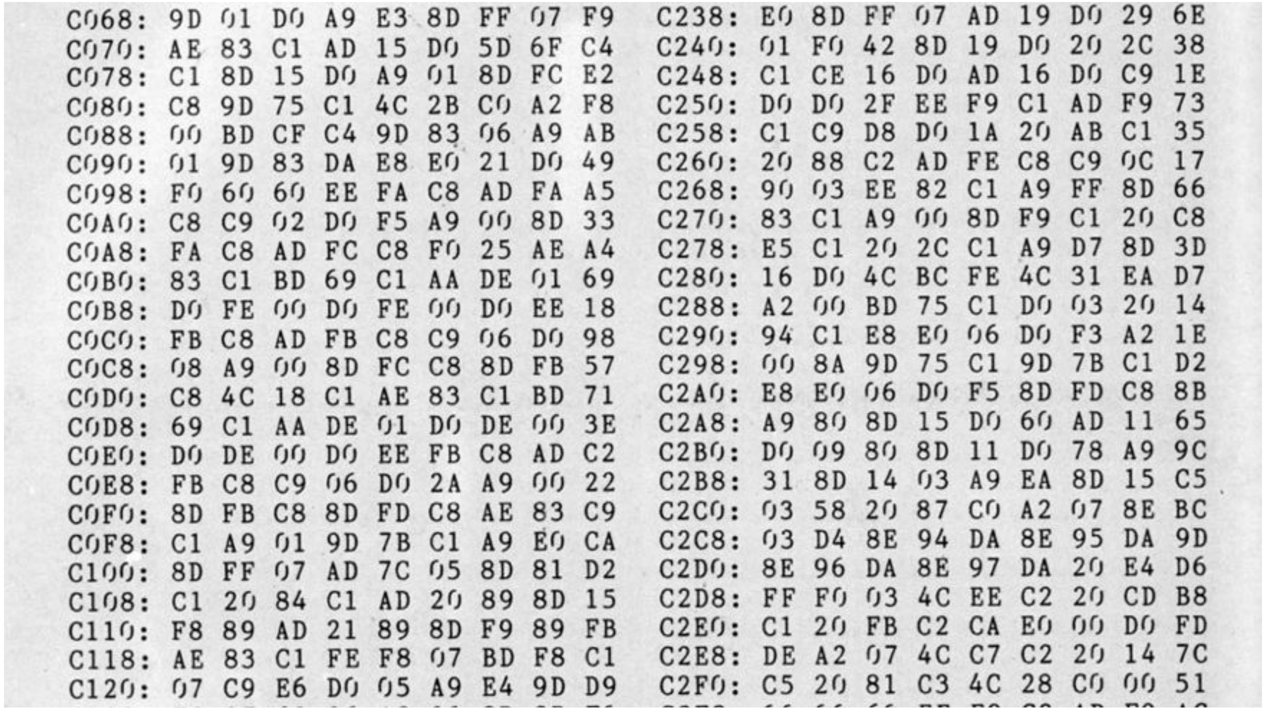
- Distinguish between code and a program.
- Define basic variables and data types in JavaScript.
- Understand the role of functions in JavaScript.



Front-End Web Development

What Is a Program?

You Might Think a Program Is Code...

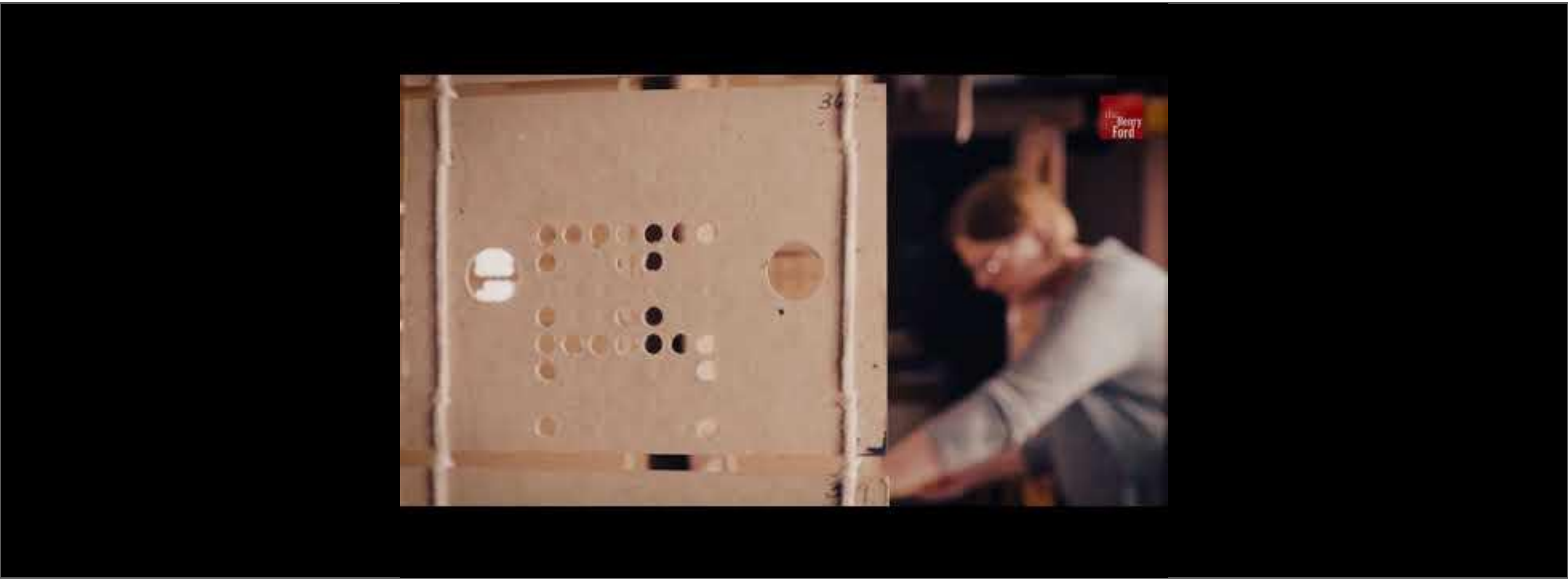


```
C068: 9D 01 D0 A9 E3 8D FF 07 F9    C238: E0 8D FF 07 AD 19 D0 29 6E
C070: AE 83 C1 AD 15 D0 5D 6F C4    C240: 01 F0 42 8D 19 D0 20 2C 38
C078: C1 8D 15 D0 A9 01 8D FC E2    C248: C1 CE 16 D0 AD 16 D0 C9 1E
C080: C8 9D 75 C1 4C 2B C0 A2 F8    C250: D0 D0 2F EE F9 C1 AD F9 73
C088: 00 BD CF C4 9D 83 06 A9 AB    C258: C1 C9 D8 D0 1A 20 AB C1 35
C090: 01 9D 83 DA E8 E0 21 D0 49    C260: 20 88 C2 AD FE C8 C9 0C 17
C098: F0 60 60 EE FA C8 AD FA A5    C268: 90 03 EE 82 C1 A9 FF 8D 66
C0A0: C8 C9 02 D0 F5 A9 00 8D 33    C270: 83 C1 A9 00 8D F9 C1 20 C8
C0A8: FA C8 AD FC C8 F0 25 AE A4    C278: E5 C1 20 2C C1 A9 D7 8D 3D
C0B0: 83 C1 BD 69 C1 AA DE 01 69    C280: 16 D0 4C BC FE 4C 31 EA D7
C0B8: D0 FE 00 D0 FE 00 D0 EE 18    C288: A2 00 BD 75 C1 D0 03 20 14
C0C0: FB C8 AD FB C8 C9 06 D0 98    C290: 94 C1 E8 E0 06 D0 F3 A2 1E
C0C8: 08 A9 00 8D FC C8 8D FB 57    C298: 00 8A 9D 75 C1 9D 7B C1 D2
C0D0: C8 4C 18 C1 AE 83 C1 BD 71    C2A0: E8 E0 06 D0 F5 8D FD C8 8B
C0D8: 69 C1 AA DE 01 D0 DE 00 3E    C2A8: A9 80 8D 15 D0 60 AD 11 65
C0E0: D0 DE 00 D0 EE FB C8 AD C2    C2B0: D0 09 80 8D 11 D0 78 A9 9C
C0E8: FB C8 C9 06 D0 2A A9 00 22    C2B8: 31 8D 14 03 A9 EA 8D 15 C5
C0F0: 8D FB C8 8D FD C8 AE 83 C9    C2C0: 03 58 20 87 C0 A2 07 8E BC
C0F8: C1 A9 01 9D 7B C1 A9 E0 CA    C2C8: 03 D4 8E 94 DA 8E 95 DA 9D
C100: 8D FF 07 AD 7C 05 8D 81 D2    C2D0: 8E 96 DA 8E 97 DA 20 E4 D6
C108: C1 20 84 C1 AD 20 89 8D 15    C2D8: FF F0 03 4C EE C2 20 CD B8
C110: F8 89 AD 21 89 8D F9 89 FB    C2E0: C1 20 FB C2 CA E0 00 D0 FD
C118: AE 83 C1 FE F8 07 BD F8 C1    C2E8: DE A2 07 4C C7 C2 20 14 7C
C120: 07 C9 E6 D0 05 A9 E4 9D D9    C2F0: C5 20 81 C3 4C 28 C0 00 51
```

Code is HOW you make a program.

A program is just a set of INSTRUCTIONS.

Could a Loom Be a Program?



How About Stir Fry?

“

The **good** news about **computers** is that **they do what you tell them to do**. The **bad** news is that **they do what you tell them to do**.

— Ted Nelson

Thinking Like a Computer

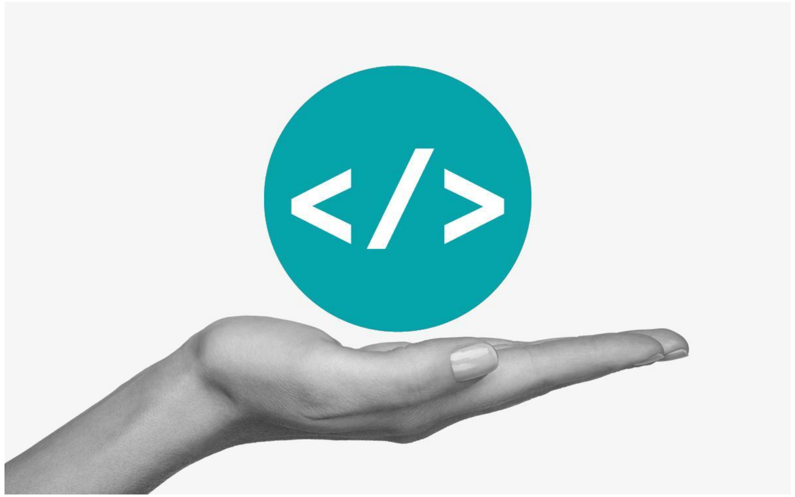
The art of programming requires understanding how a computer thinks:

It only knows what you tell it…	…but it will remember what it's been told.
It only understands a very limited set of phrases (syntax)…	…but you can teach it a lot by combining these basic phrases together.
It will always do what you say…	…but not necessarily what you meant.
It has no understanding of context…	…but it's not shy about saying when it doesn't understand you (error messages).

Using Pseudocode

Pseudocode uses the structure of code without having to worry about the actual coding language (syntax) of the instructions.

Pseudocode is an attempt to break down a larger process into the smallest imaginable component steps.



A Stir Fry Program

spices = soySauce + riceVinegar + sugar
oil = 1.5 ounces
brownRice = 1.5 cups
broccoli = 2 cups
shrimp = .5 pounds

```
    stirFry {
        cook brownRice
        whisk spices
        heat oil in pan
        add broccoli
        cook shrimp
        add spices
    }
```

A Stir Fry Program

spices = soySauce + riceVinegar + sugar
oil = 1.5 ounces
brownRice = 1.5 cups
broccoli = 2 cups
shrimp = .5 pounds

```
stirFry(item1, item2, item3, item4, item5) {
     cook(item1)
     whisk(item2, item3)
     heat(item3)
     add(item3, item4)
     cook(item5)
     add(item2)
     }
stirFry(brownRice, spices, oil, broccoli, shrimp)
```

The ingredients act as **variables** being passed into the `stirFry` **function**.

Recipe verbs are **functions** you can use with the ingredient variables.

Variables

The Nouns of Programming

Pieces of data in a program that can used and changed many times.

These are used for anything you need the computer to remember or keep track of for later. If you would write it down, it's a variable.

Functions

The Verbs of Programming

Modular, reusable containers for an instruction set.

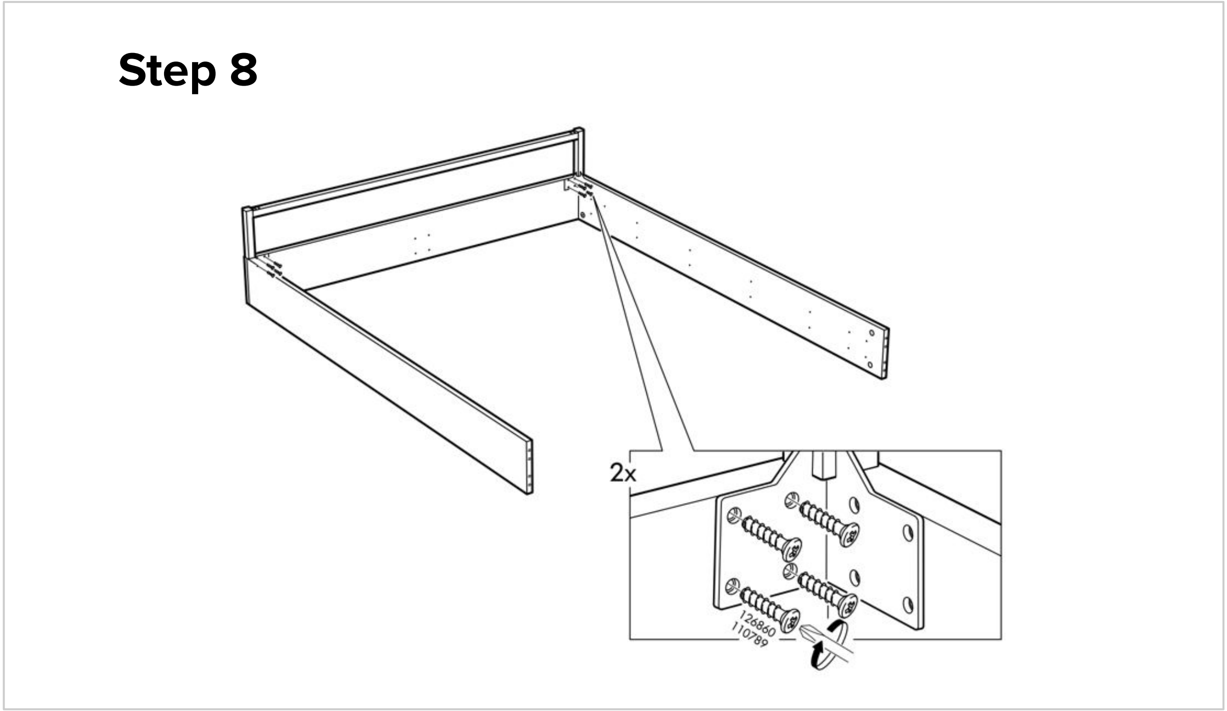
These allow us to define building-block procedures that provide readable instructions for complex operations.

Anyone have an IKEA bed?

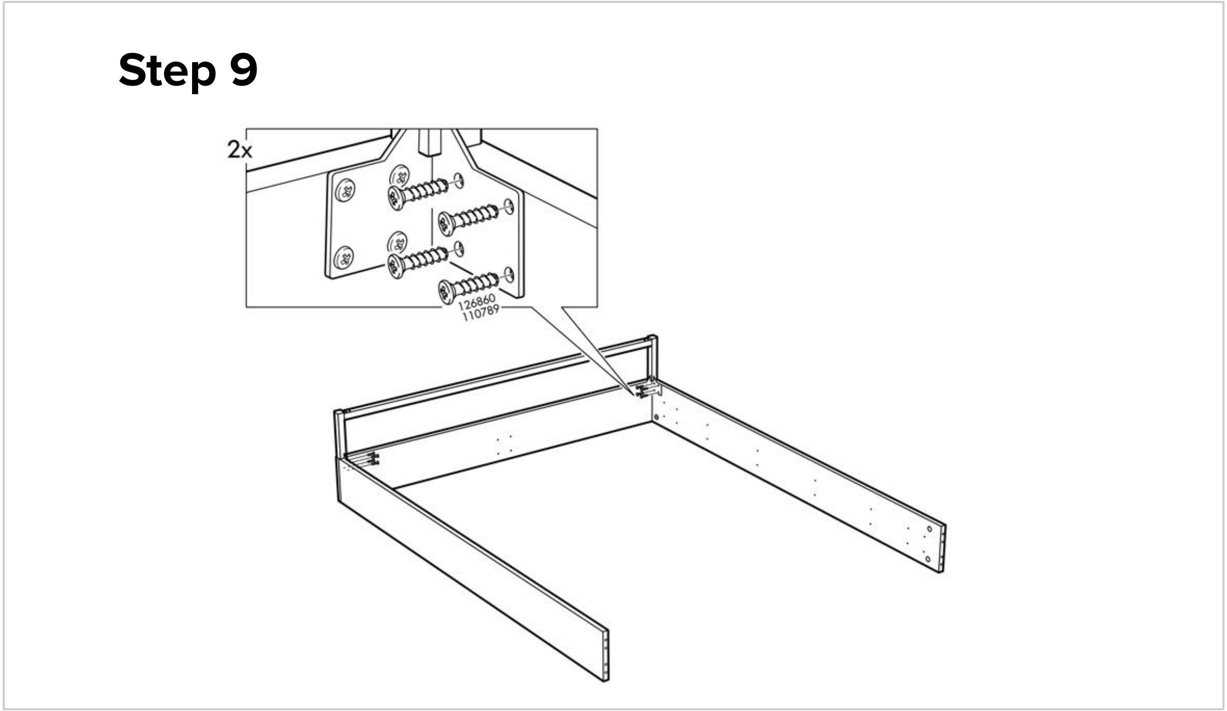
They require **instructions** to build.

Instruction manuals are also **programs**.

Can You Describe Step 8?



How Is It Different Than Step 9?

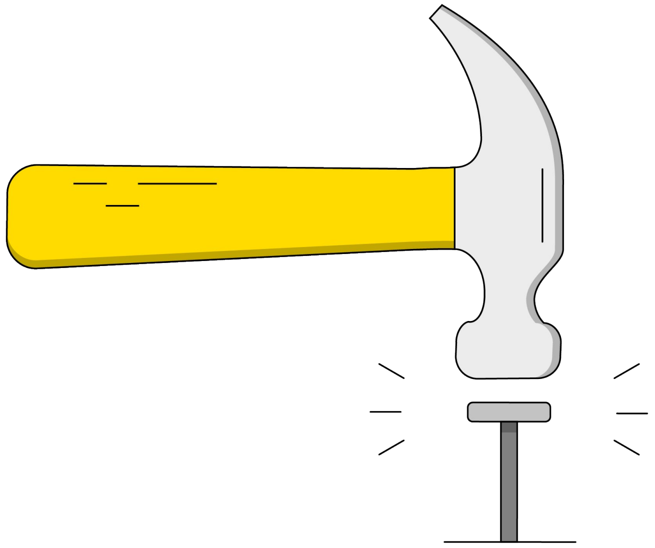


Break It Down

We need to put together the headboard and sideboards for a bed.

But a computer has no context for what a bed is! Keep that in mind.

It's going to take four sets of four screws and two metal corner brackets. So we've got many sets of very similar instructions.



In English...

Step 8: Get four screws and a corner bracket. Bring together the headboard and sideboard. Screw the bracket into the headboard. Repeat the process for both sides of the bed.

Step 9: Get four more of the same screws and screw the bracket into the sideboard of the bed. Repeat the process for both sides of the bed.

Group Exercise:

Every Day I’m Pseudocoding

30 minutes

In groups, write out pseudocode for an activity of your choice. Think about the repeated processes involved in that activity — those are your functions!

What items does the activity require? Those are your variables!

Remember, computers need instructions broken down into the tiniest steps imaginable using very direct language and clearly defined parts.

Programs don't get more complicated because of code, they get more complicated because of the logic behind them.

Front-End Web Development

JavaScript: The Basics

Guided Walk-Through:

JavaScript Reference Desk

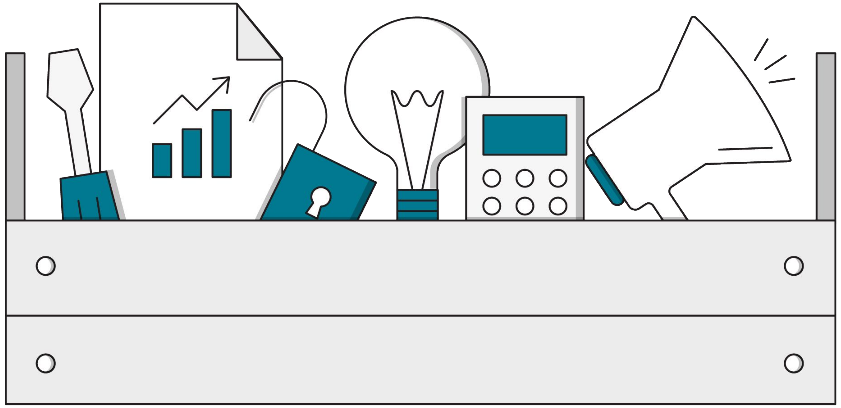
You can reference the JavaScript concepts with examples in this CodePen:

Reference code:

https://codepen.io/GAmarketing/pen/rNaezQM

Basic Data Types

- Numbers
- Strings
- Booleans
- Null/undefined



Numbers

- You've probably seen these before: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.
- They are used mathematically throughout JS code, so normal math rules apply.
- They are paired with operators: `+`, `-`, `*`, and `/`.
  - `10 * 10 ⇒ 100`
  - `8 - 4 ⇒ 4`
  - `49 / 7 ⇒ 7`
- They can also include floating point numbers — i.e., decimals or floats.
  - `8.99 + 2 => 10.99`

Strings

- String means **text** — that's it!
  - `"It is a beautiful evening";`
  - `"Is it really Monday?";`
  - `"I am feeling good today!";`
- With JS, you can merge strings using the `"+"` operator. This is called **“string concatenation.”**
  - `"You want " + "to go " + "eat on 14th?";`
- You can think of a string as a collection of **characters** tied together.

Booleans

- Booleans represent the logical concept of **true or false**.
  - Other data values can be converted to Booleans for logical analysis.
  - `0`, `-0`, `null`, `NaN`, `undefined`, or the empty string (`""`) are `false`.
  - All other values will be converted to `true` — if it exists, it's "truthy."

```
Boolean("Jack Nicholson");
⇒ true

Boolean("1979");
⇒ true

Boolean(0);
⇒ false
```

Null/Undefined/NaN

- These values denote the lack of value in JavaScript.
  - **`null`** specifically suggests nothing — i.e., certain not to be anything.
  - **`undefined`** suggests a variable will be given a value later but not yet.
  - **NaN** means “not a number,” usually because your math has gone wrong.

```
let lysine;
console.log(lysine);
⇒ undefined

console.log(9 * null);
⇒ 0

console.log("five" * 5);
⇒ NaN
```

Front-End Web Development

Doing Stuff With Data

Variables

- Variables let us store data in a program.
- Declare variables using the **`let`** or **`const`** keywords.
    - **`const`**, short for “constant,” is for variables that won’t be assigned new values.
    - **`let`** is for variables that will be given new values, for example when using math.
- You refer to variables by using their names anywhere in your program.

```
let flyingMonkeys = 5;
let tiredGoats = 7

flyingMonkeys + tiredGoats
⇒ 12
```

Variable Naming

- Names should be easily understood.
- No spaces allowed.
- You *can* use "_", but **don't do it**.
- Use camelCase: itWorksLikeThis.

```
let howManyCoasters = 18;
const midWeek = "Wednesday";
```

Variable Reassignment

```
let x = 18;
console.log(x);
⇒ 18

x * 2;
console.log(x);
⇒ 18

x = x * 2
console.log(x);
⇒ 36
```

Note: If you'd used **`const`** instead of **`let`** to declare the variable, the last part would throw an error!

Messages and Alerts

```
// Writes to webpage
document.write("Yes!");

// Writes to console
console.log("Houston, do you copy?");

// Makes popups happen
alert("You destroyed the computer again.");
```

Solo Exercise:

Data Types and Logging in JS

30 minutes

Practice using console.log and basic variable operations in JavaScript. There's more than one way of doing practically anything in JS– google and Stack Overflow will be your best friends in JS land!

Starter code:

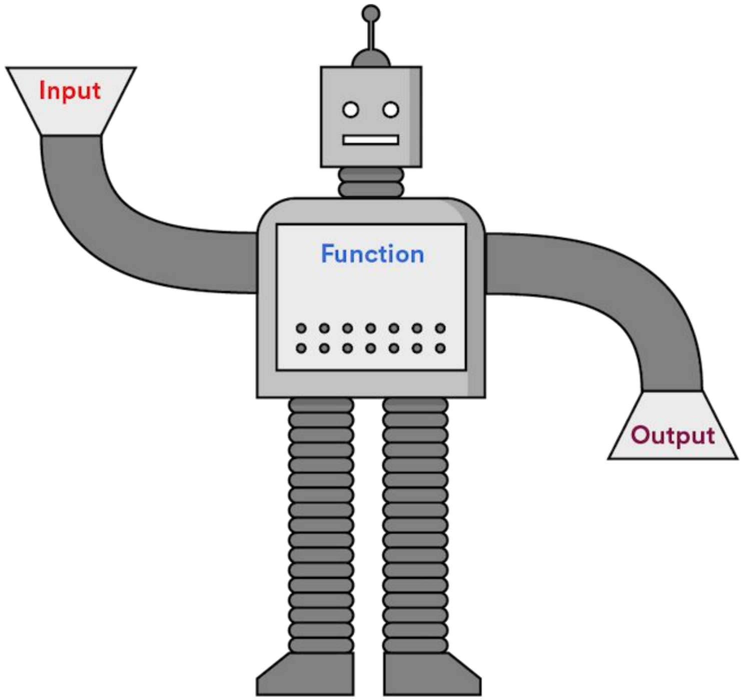
https://codepen.io/GAmarketing/pen/wvwJQNz

Solution code:

https://codepen.io/GAmarketing/pen/rNByQPv

Functions

- **Functions** are chunks of code that are grouped and execute together, like a modular program within a program.
- A function takes input, performs logic, and returns output.



Functions

- Reusable functions need names and follow naming rules that are identical to those of variables.
- They must be called or “**invoked**” to execute and return a value.
- They accept input, called **parameters**, that can be used inside the function.

```
function greetings(name) {
  console.log(“Hello there, “ + name);
}

greetings(“Dave”);
```

Key Takeaways

Computers Think in Small Steps

- Use pseudocode to plan out steps of a program.
- You can define functions to encapsulate common procedures.
- Use variables for anything you need the computer to keep track of.

For Next Time

JavaScript and the DOM

- Use JavaScript to interact with elements on the page.
- Respond to user actions by setting event listeners.