Introduction:

I have a JavaScript book in front of me, it has about 600 pages. Most of it is just text. Is like reading a history book. Is boring. This book will teach you the fundamentals: loops, variables, arrays, and much more. These concepts have not changed that much for the last 20 years.

Why is my book different?

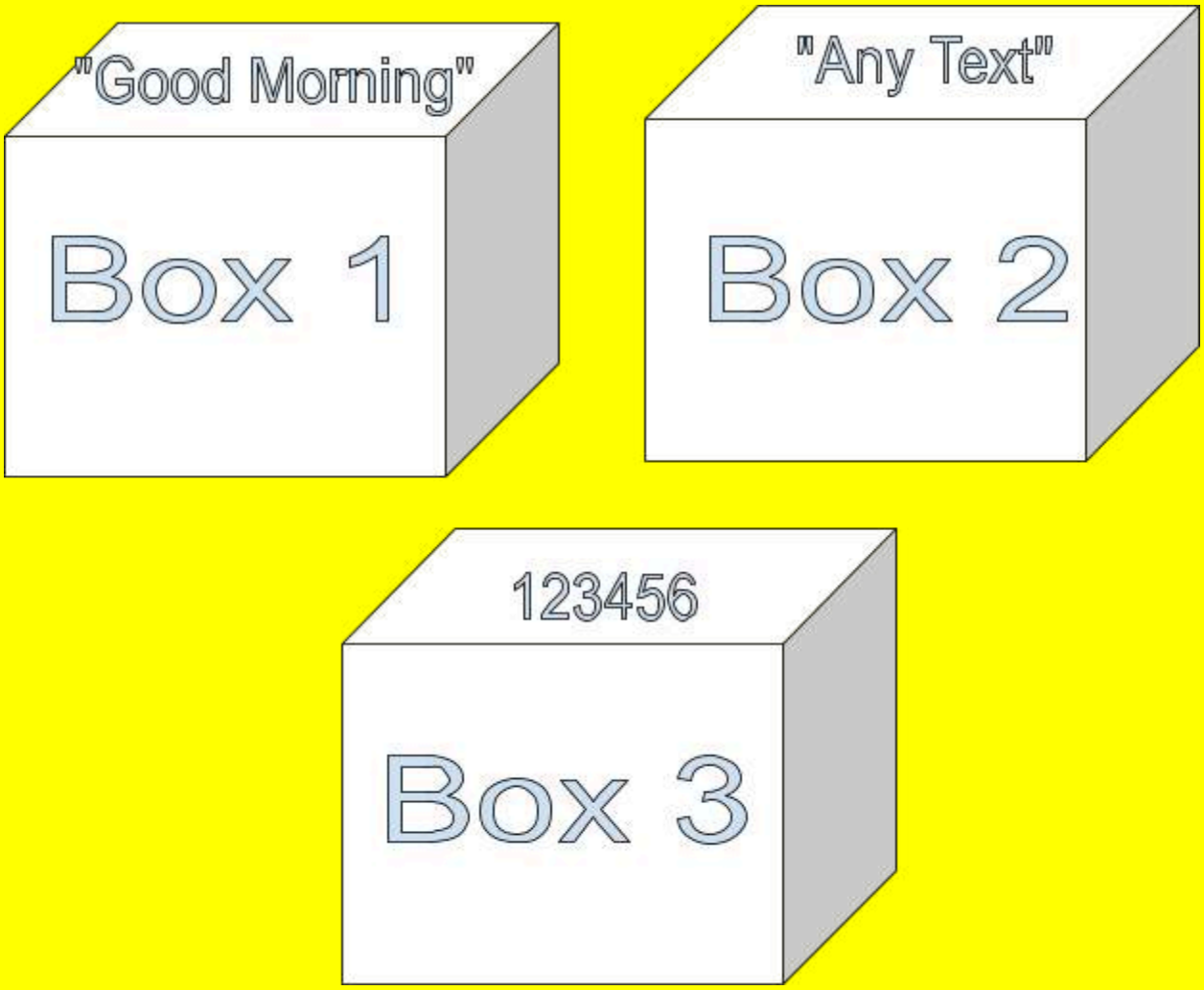
I have provided you with real life examples. Most important thing, those examples are strictly designed so you can start to see patterns in JavaScript. Once you start to see a pattern, you will have an “AHA” moment. Because you will be able to apply those patterns somewhere else.

Last thing, writing code is an ART. Some people can write a full page of code, and some people can write the same page of code in just a few lines. Keep an open mind, and learn to love the process.

Have fun! Allow yourself to have fun especially when you make a mistake. Don’t take it too seriously. The road will get difficult, the more you enjoy learning programming, the easier it will get.

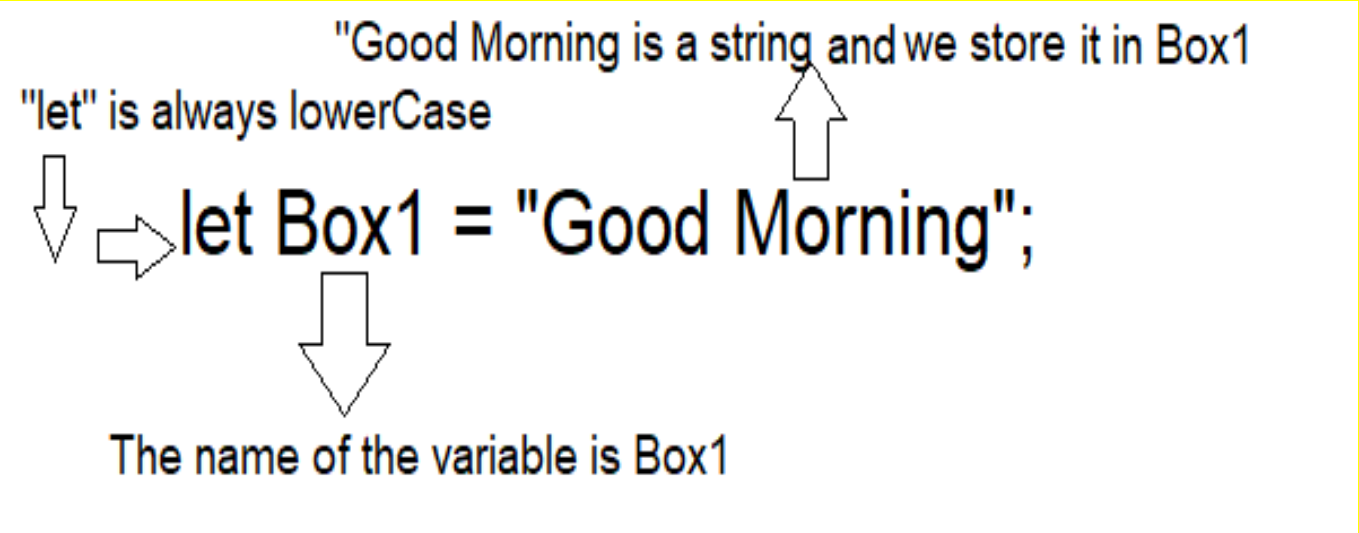
Remember, one day at the time. You are becoming a better you, if that’s not enough. You can actually make a difference in this world with your programming skills. Enough, let’s start coding >>>

{Hour 1} - Variables



##Section 1.1: The Anatomy of a Variable.

We used variables in JavaScript to store numbers or strings. We use **let**,**const**, or **var** to tell JavaScript that we need to create a variable. For the rest of the book we are only going to be using **let** for simplicity purposes.



Section 1.2: With console.log() is how we output variables, is how we find out if our variables are working. Be patient if you don't get it yet, you will be using it a lot.

Syntax:

```
console.log(variable)
```

Example 1:

```
// Declare a variable 'box1' and assign the value "Good
Morning" to it
let box1 = "Good Morning";

// Print the value of 'box1' to the console
console.log(box1);

// Expected output: "Good Morning"
```

Example 2

```
// Declare a variable 'box2' and assign the value "Good
Afternoon" to it
let box2 = "Good Afternoon";

// Print the value of 'box2' to the console
console.log(box2);

// Expected output: "Good Afternoon"
```

Section 1.3: You can use single quotes (‘) or double quotes (“) to represent data. Some people will use single quotes and others will use double quotes, whatever you choose be consistent.

Example 1

```
let withDoubleQuotes = "This is a string with double quotes.";
let withSingleQuotes = 'This is a string with single quotes.';

console.log(withDoubleQuotes);
console.log(withSingleQuotes);
```

Example 2

```
let stringWithQuotesInside = "He said, 'Hello!'";

// or

let stringWithQuotesInside2 = 'She replied, "Hi there!"';
```

Section 1.4: We remove the single quotes (‘), or (“) double quotes when we have numbers.

Example 1

```
// Declare a variable 'num1' and assign the value 5
let num1 = 5;

// Declare a variable 'num2' and assign the value 7
let num2 = 7;

// Perform addition using the variables 'num1' and 'num2' and
store the result in 'sum'
let sum = num1 + num2;

// Print the value of 'sum' to the console
console.log(sum); // Expected output: 12
```

Example 2

```
// Declare a variable 'a' and assign the value 10
let a = 10;
// Declare a variable 'b' and assign the value 3
let b = 3;
// Perform multiplication using the variables 'a' and 'b' and
store the result in 'product'

let product = a * b;
// Print the value of 'product' to the console
console.log(product); // Expected output: 30
```

Section 1.5: Single Line Comments

We add a single line comment in JavaScript by adding ‘//’ into your code. As the code gets more complicated, those ‘//’ are like little messages explaining what you're trying to do with your code. Your comments will be ignored by the JavaScript compiler.

Example 1

```
//version 0.2
//unhackable version
//this is a comment
//<script> second comment.

let box1 = "Good Morning";

console.log(box1);

//this line will not affect your code
//alert("test")
```

Example 2

```
//version 0.2
// box2 holds “Good Morning”;
let box2 = "Good Morning";
console.log(box2);

//this line will not affect your code
//alert("test")
```

Section 1.6: Multi-Line Comments

In order to add a multi-line comment, you add a (“/*”) at the beginning and a (“*/”) at the end.

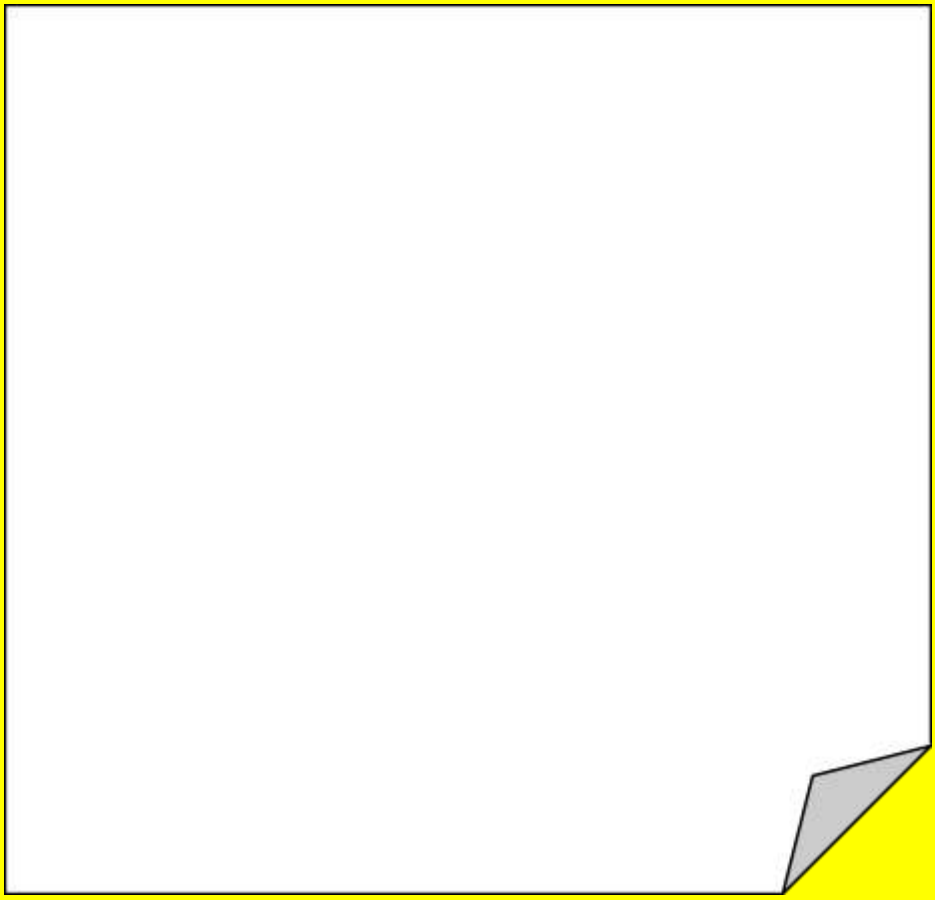
Example 3

```
/*
  This is a multi-line comment.
  It can span several lines and is often used for longer
explanations or notes.
*/

let anotherVariable = 42;
```

Example 4

```
/*
  This is a multi-line comment.
  It can span several lines and is often used for longer
explanations or notes.
*/

let anotherVariable = 42;

//anotherVariable holds the amount of liquid
/*
  This is a multi-line comment.
  It can span several lines and is often used for longer
explanations or notes.
*/
```

{Hour 2} - `{Template Literals}`



Section 2.1: Concatenation

The plus sign (“+”) operator is utilized for the purpose of outputting two or more strings together."

Example 1:

```
let firstName = "John";
let lastName = "Doe";

// Concatenation using the + operator
let fullName = firstName + lastName;
console.log(fullName);
// Output: JohnDoe
```

Example:2:

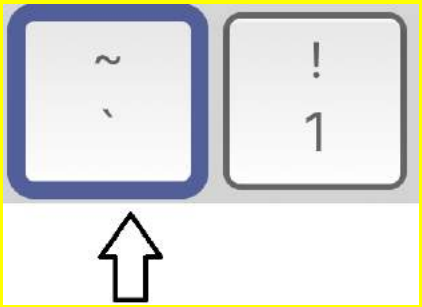
```
let firstName = "John";
let lastName = "Doe";

// Concatenation using the + operator
let fullName = firstName + " " + lastName; //we used “ “ to add space
// Output: John Doe
```

Section 2.2: Template literals are defined by enclosing them in backticks (`). It offers a more visually appealing option than concatenation.

Syntax:

```
`Hi ${anyVariable}, anytext `
```

You will find the backticks (`) next to the 1 in your keyboard.



Example 1:

```
let name = "Alice";
let greeting = `Hello, ${name}!`;
console.log(greeting); // Output: Hello, Alice!
```

Example 2:

```
let firstName = "John";
let lastName = "Doe";
let fullName = `${firstName} ${lastName}`;
let greeting = `Welcome, ${fullName}!`;
console.log(greeting);
```

Example 3:

```
let a = 5;
let b = 10;
let sum = `The sum of ${a} and ${b} is ${a + b}.`;
console.log(sum); // Output: The sum of 5 and 10 is 15.
```

Example 4:

```
let title = "Dynamic Title";
let content = "Some dynamic content.";
let html = `
<div>
  <h1>${title}</h1>
  <p>${content}</p>
</div>
`;
console.log(html);
```