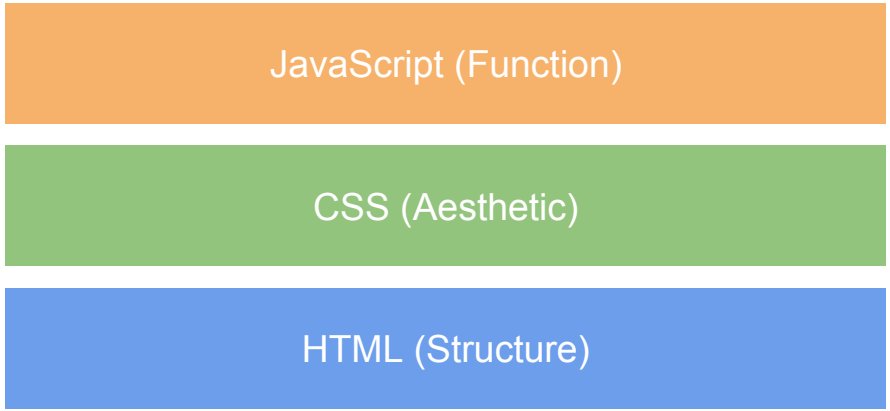
JavaScript

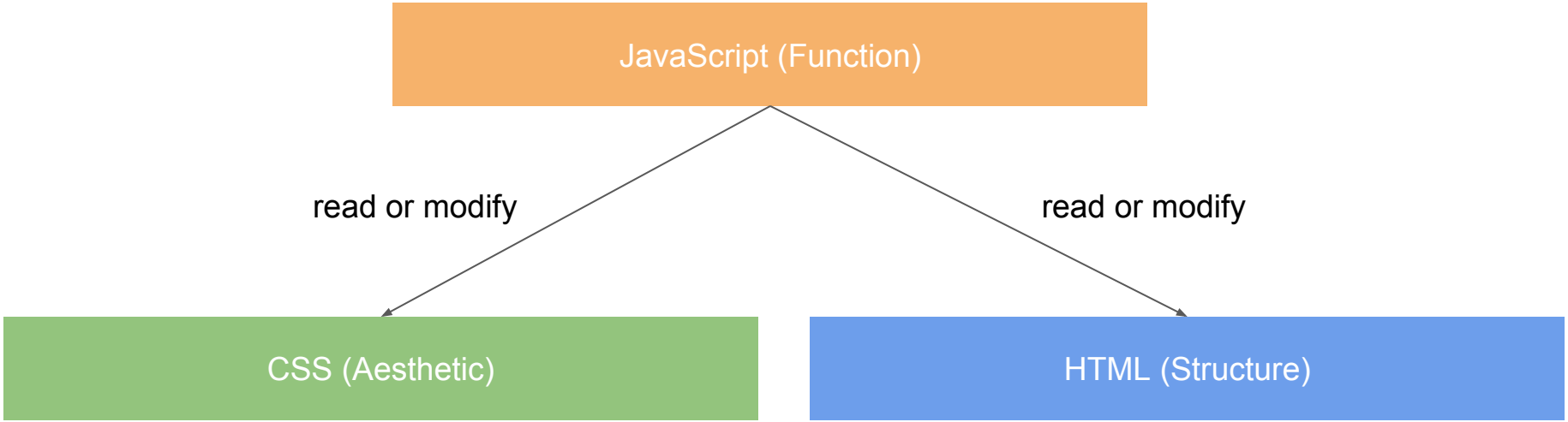
Recap

- HTML
- CSS
- Design Thinking
- JavaScript

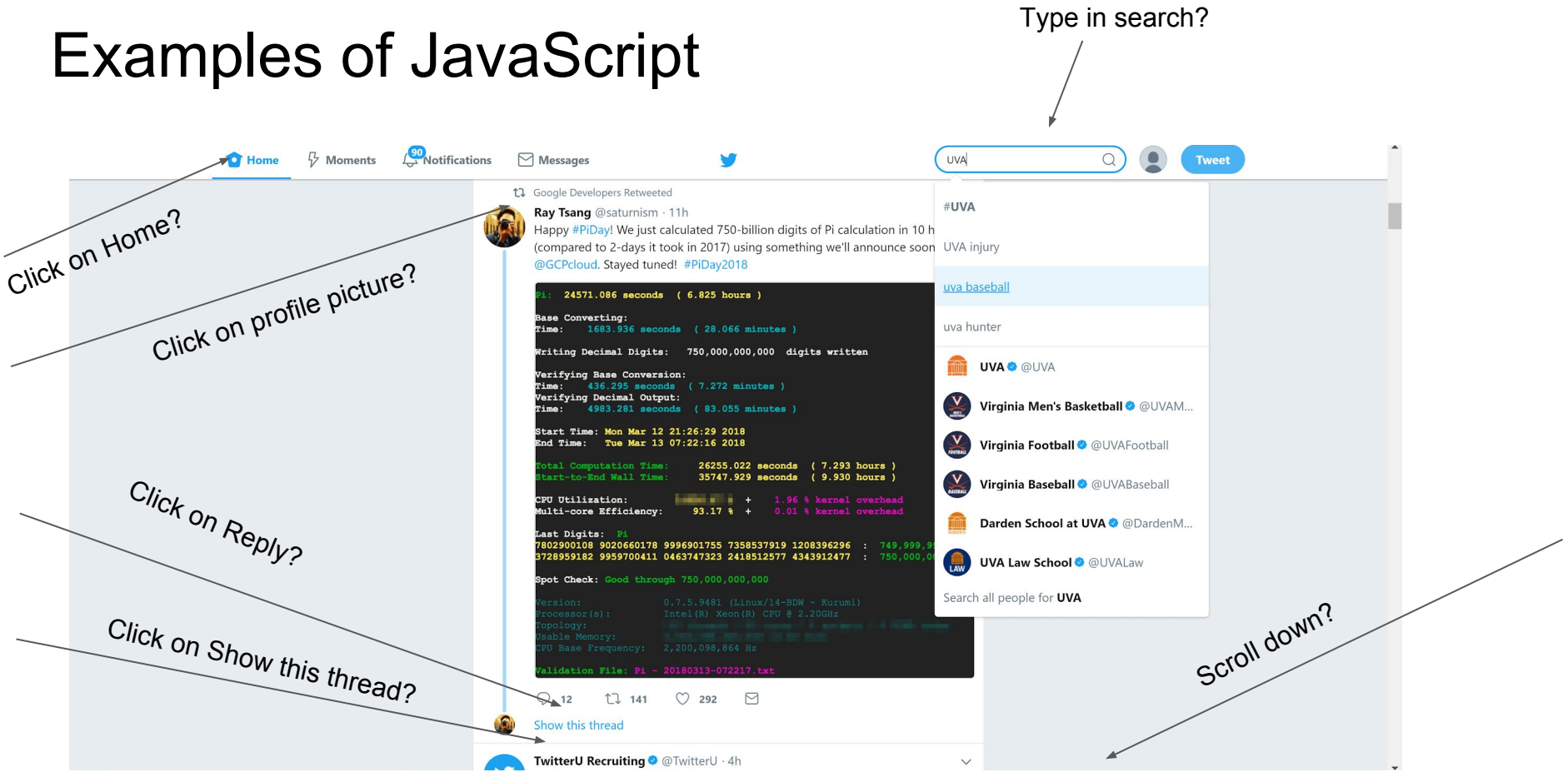
Where does JavaScript fit in?



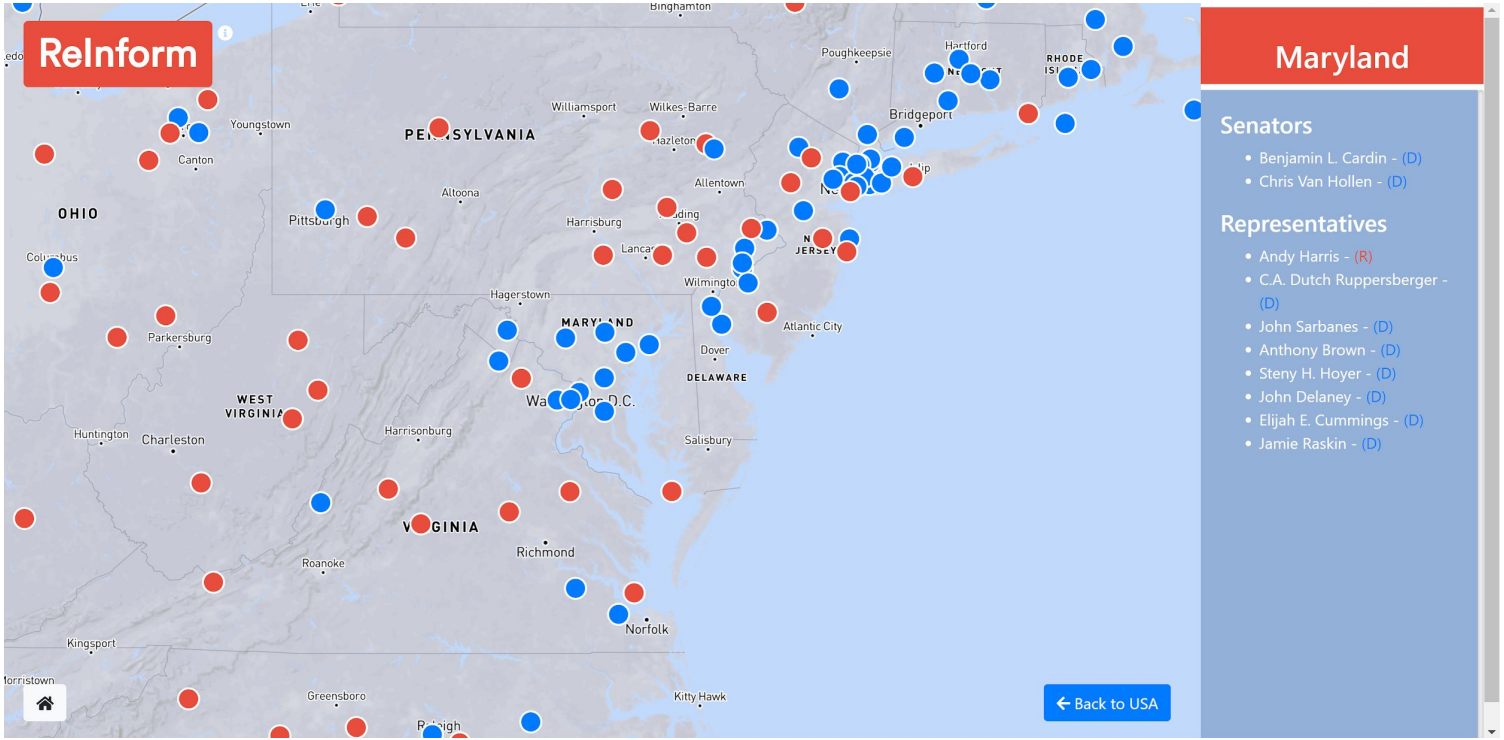
Where does JavaScript fit in?



Examples of JavaScript

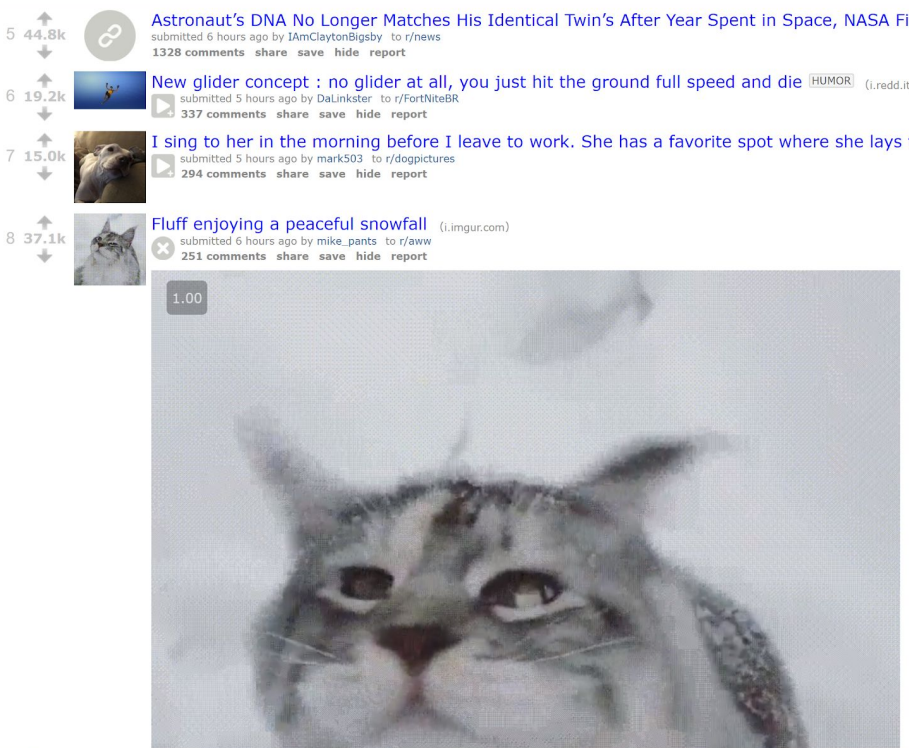


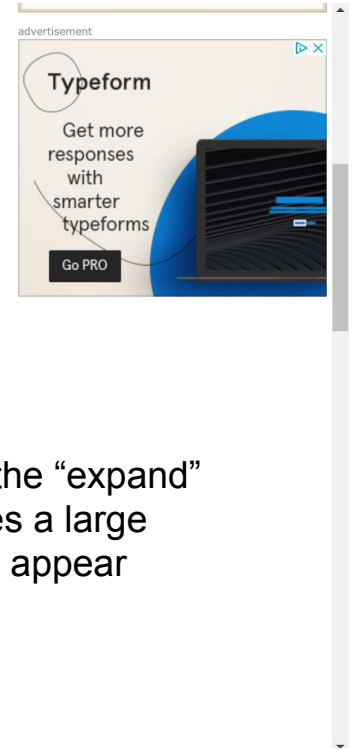
Examples of JavaScript



Reinform.me

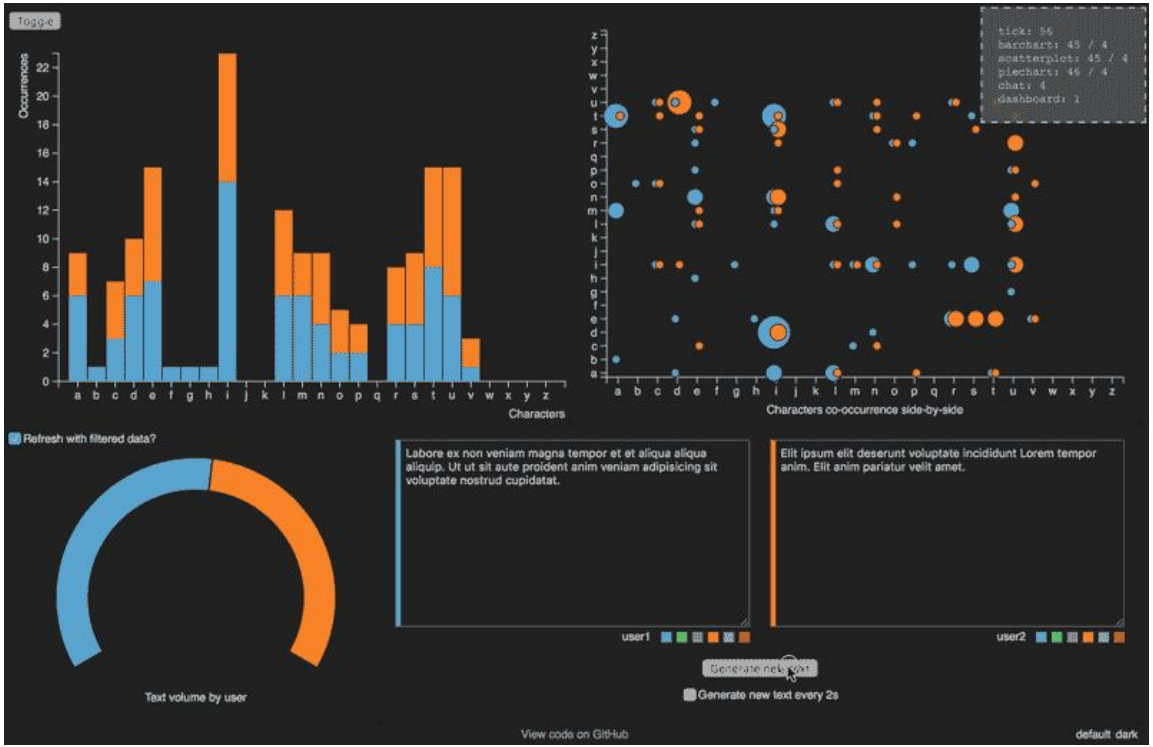
Examples of JavaScript





Clicking on the "expand" button makes a large image/video appear

Examples of JavaScript



https://rd3.now.sh/

Where is JavaScript written?

```
<!DOCTYPE html>
<html>
    <head>...</head>
    <body>
        <p>...</p>
        <script>
            // javascript goes here
        </script>
    </body>
</html>
```

Where is JavaScript written?

- Can also be written in a separate .js file and imported in

```
<!DOCTYPE html>
<html>
    <head>
        <script src="myscripts.js"></script>
    </head>
    <body>...</body>
</html>
```

Variables

- Store information
  - numbers, strings, arrays, objects, functions
- Variable names
  - must be unique
  - can contain letters, digits, underscores and dollar symbols
  - cannot start with a digit
  - are case sensitive
  - cannot be a keyword

Variables

- Declaration

  ```
  var x;  // undefined
  ```

- Initialization

  ```
  x = 3;
  ```

- Both

  ```
  var y = “hi”;
  ```

Variable Types

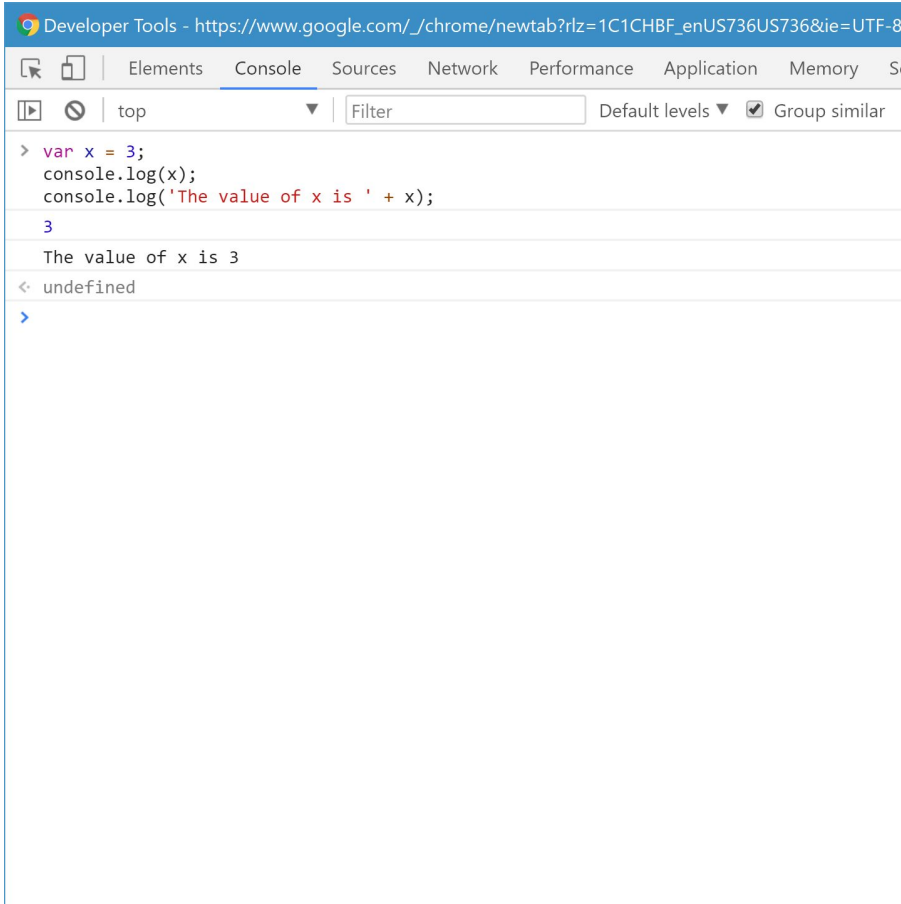
```
var anumber = 3;
var anothernumber = 3.0;
var astring = “hello”;
var anotherstring = ‘world’;
var anarray = [];
var anotherarray = [3, “hello”, anotherstring];
var anobject = {};
var anotherobject = {“key”: “value”};
```

Operators

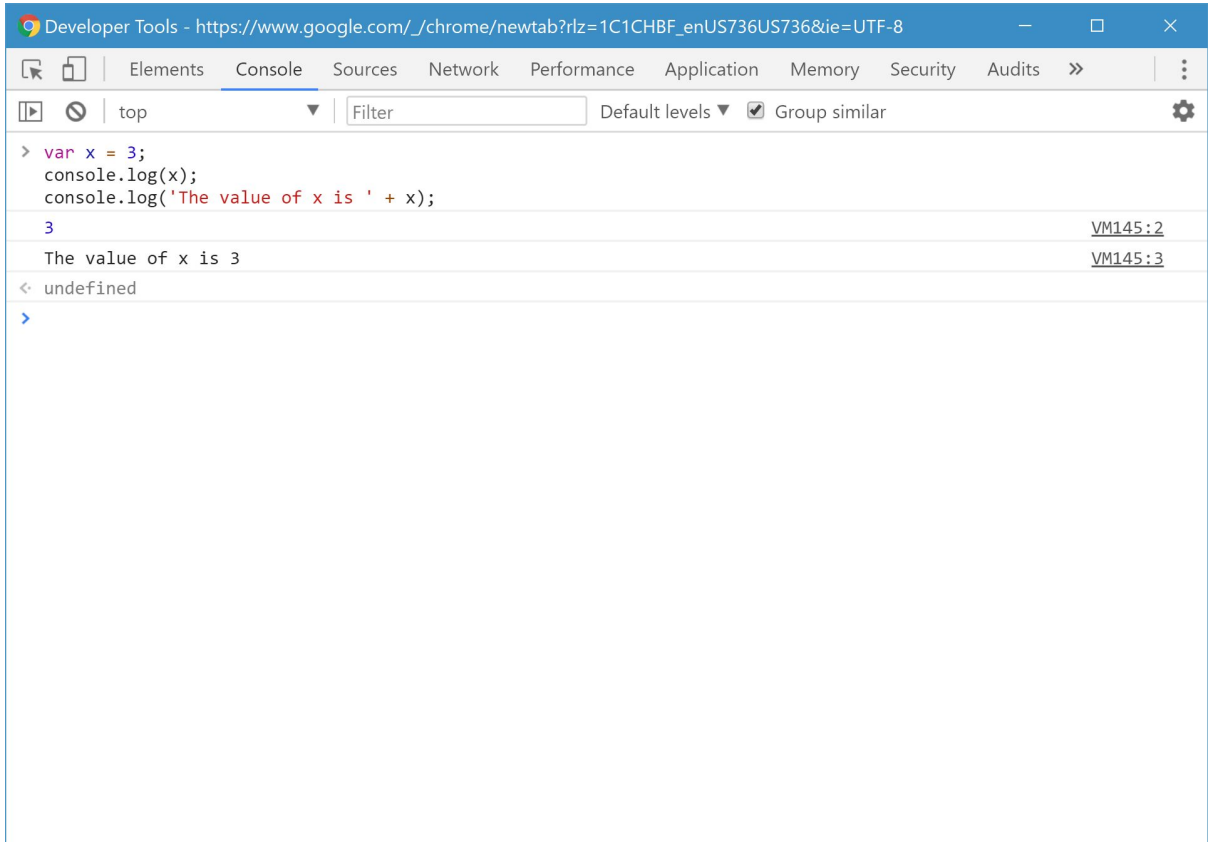
```
var x = 5;  // 5

x = x + 1;  // 6

x += 1;  // 7

x -= 1;  // 6

x *= 4;  // 24

x /= 3;  // 8

x %= 5;  // 3
```

Printing variables

```
var x = 3;

console.log(x);

console.log('The value of x is ' + x);
```



Recap: Chrome Developer Tools

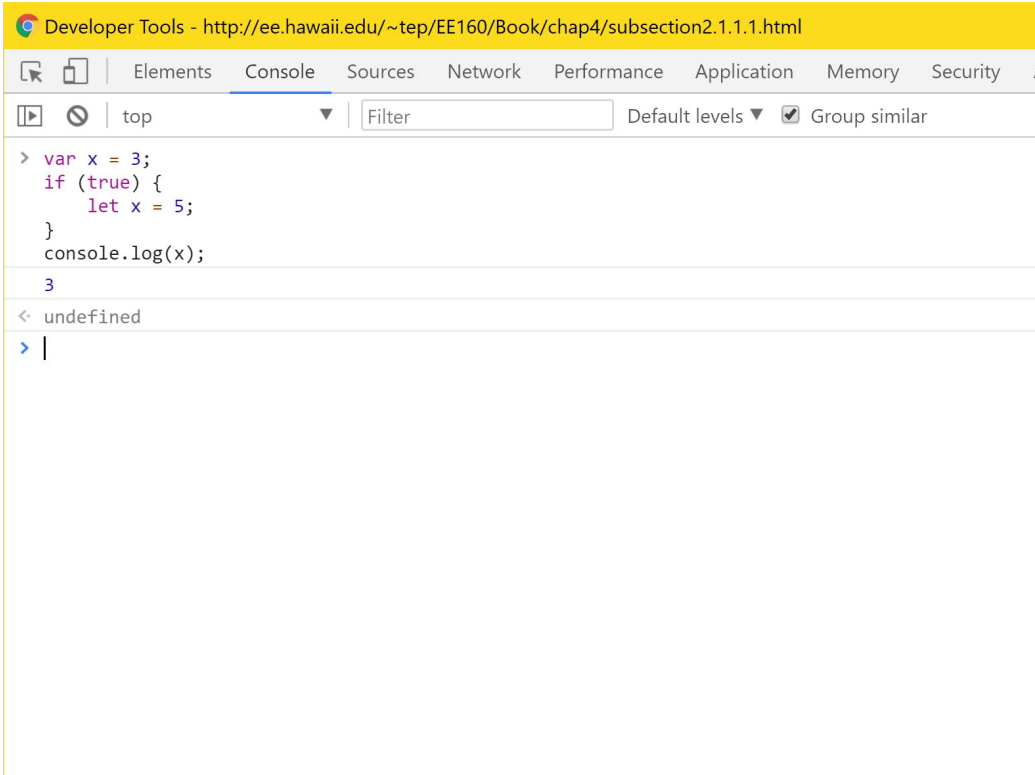


More on Variables

let declares a variable in the local scope

var declares a variable in the global scope

```
var x = 3;
if (true) {
    let x = 5;
}
console.log(x);
```



If statements

```
if(condition) {
        // then do this
}
```

But first, conditions...

<, <=, >, >=

== vs. ===

!= and !==

&&, ||

```
"" == 0  // true
0 == "0"  // true
"0" == ""  // false

"" === 0  // false
0 === "0"  // false
"0" === ""  // false
```

If statements

```
if(hungerLevel >= 5) {
        orderPizza();  // function call
}
```

If statements

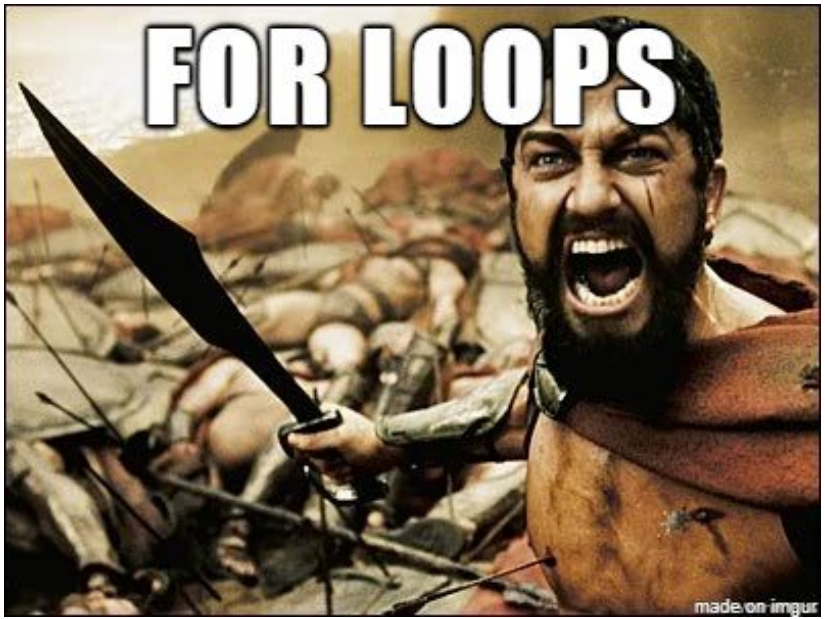
```
if(hungerLevel >= 5) {
        orderPizza();  // function call
} else if(hungerLevel >= 3) {
        goToNewcomb();
}
```

If statements

```
if(hungerLevel >= 5) {
        orderPizza();  // function call
} else if(hungerLevel >= 3) {
        goToNewcomb();
} else {
        subsist();
}
```

Loops

- **for**
- for-in
- **while**
- do-while



For-loops

```
for (statement 1; statement 2; statement 3) {
        // code you want looped
}
```

statement 1: executes before the loop starts

statement 2: defines the condition for looping

statement 3: executes for every iteration of the loop

For-loops

```
for(let i = 0; i < 6; i++) {
        console.log(i); // do something using our index value
}
for(let r = 0; r < 6; r++) {
        for(let c = 0; c < 6; c++) {
                console.log(r + c); // for-loops can be nested
        }
}
```

While loops

```
var check = true;
while(check) {
	check = checkIfSomethingIsTrue(check);
}
```

While loops

- Can also be implemented using a “base case”

```
var i = 0;
var check = true;
while(check) {
	if(i > 6)
		break;
	check = checkIfSomethingIsTrue(check);
}
```

Functions

```
var i = myFunction(3,4);
console.log(i);  // This will print out the number 12

function myFunction(p1, p2) {
    return p1 * p2;  // The function returns the product of p1 and p2
}
```

Functions

Normal function definition.

```
function foo(a, b) {
      return a + b;
}
console.log(foo(3, 5));  // will print 8
```

Functions can be stored in variables.

```
let bar = function(a, b) {
  return a + b;
}
console.log(bar(3, 5));  // will print 8
```

Functions

Just a heads up, functions can also be defined in some weird ways.

```
let baz = (a, b) => {
   return a + b;
}
console.log(baz(3, 5));  // will print 8
```

Linking Javascript to HTML

We discussed HTML's tree structure earlier

When an HTML document is loaded into a web browser, it becomes a document object

- Document object is the root node of the HTML document
- "Owner" of all other nodes
  - element nodes, text nodes, attribute nodes, and comment nodes
- Provides properties and methods to access all node objects from within JavaScript
- Very useful for dynamic programming

https://www.w3schools.com/js/tryit.asp?filename=tryjs_intro_inner_html