

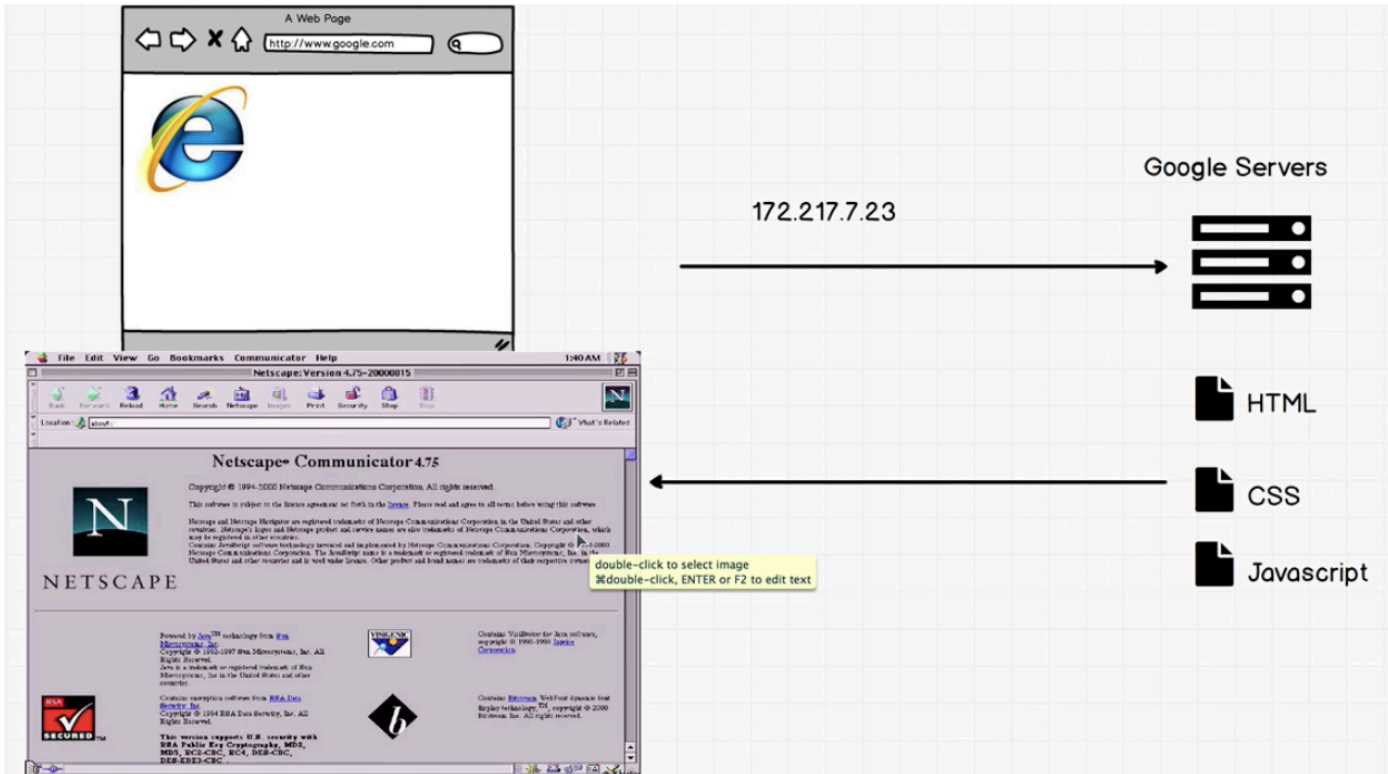
# **JavaScript: Types & Comparisons**

The  
Complete  
Web  
Developer in  
2018

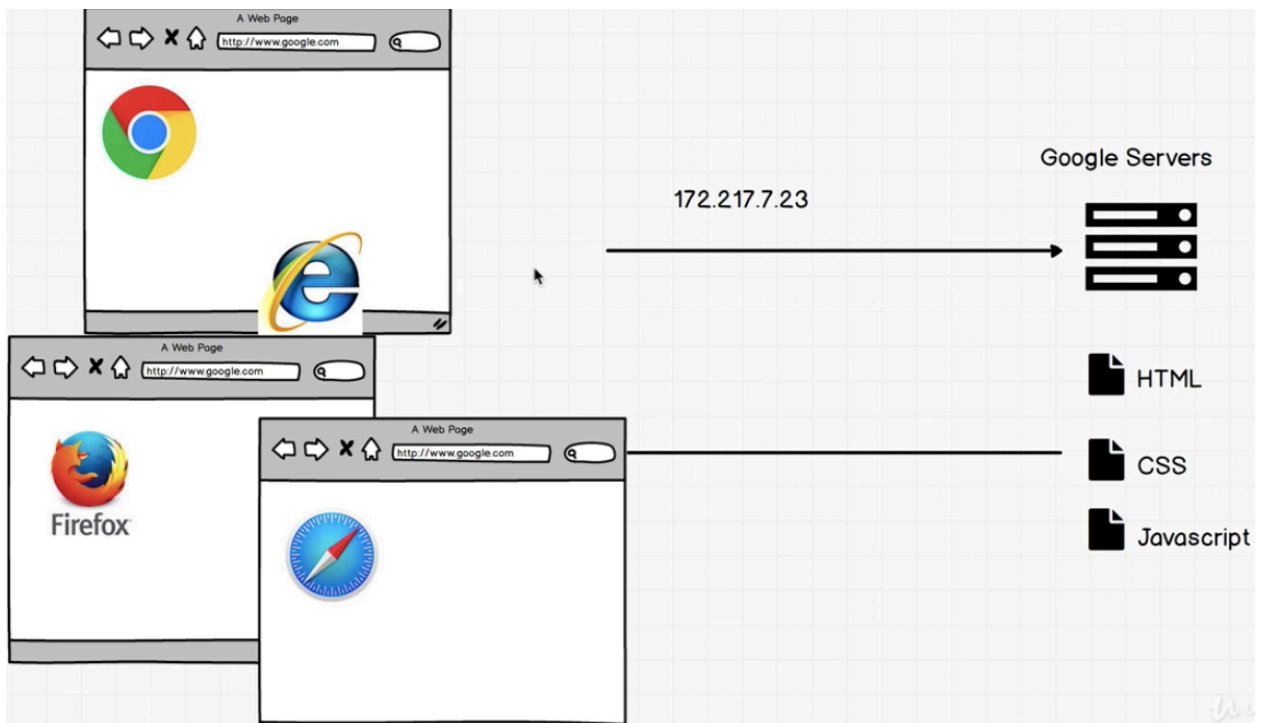
The Complete Web Developer in 2018  
Zero to Mastery  
Andrei Neagoie  
Lecture Notes by Stephanie

# Javascript

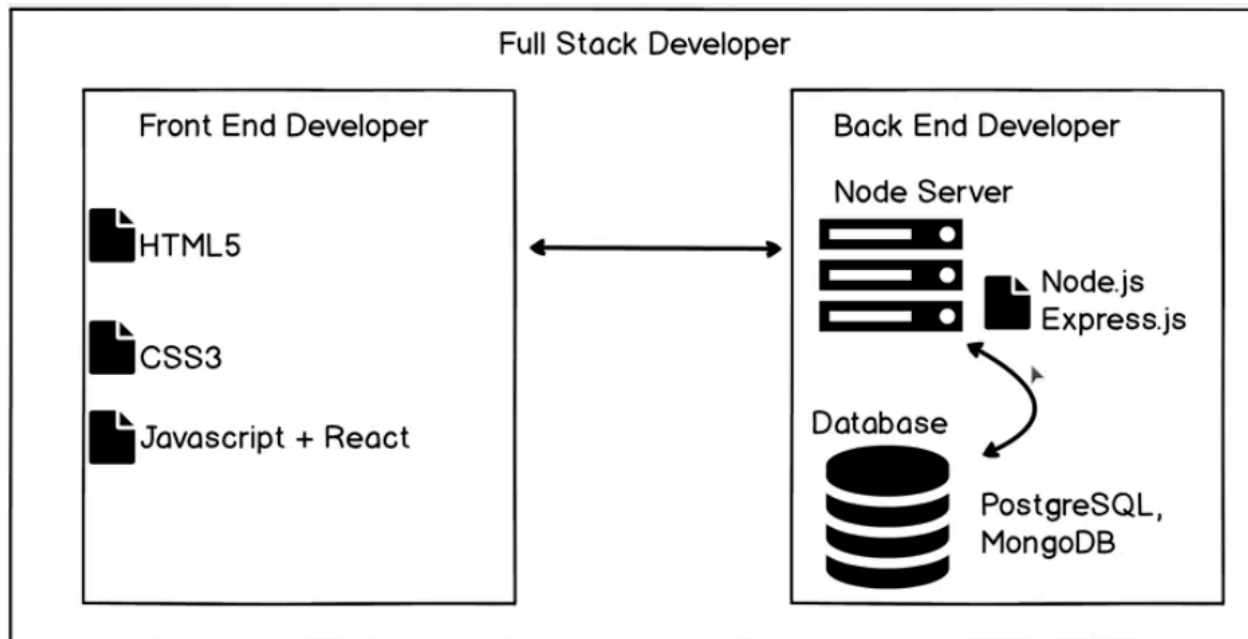
originally Netscape - Javascript performs *actions*



Now- web, apps, VR, drones, robotics



Javascript used in Node.js, Express.js, react (backend)

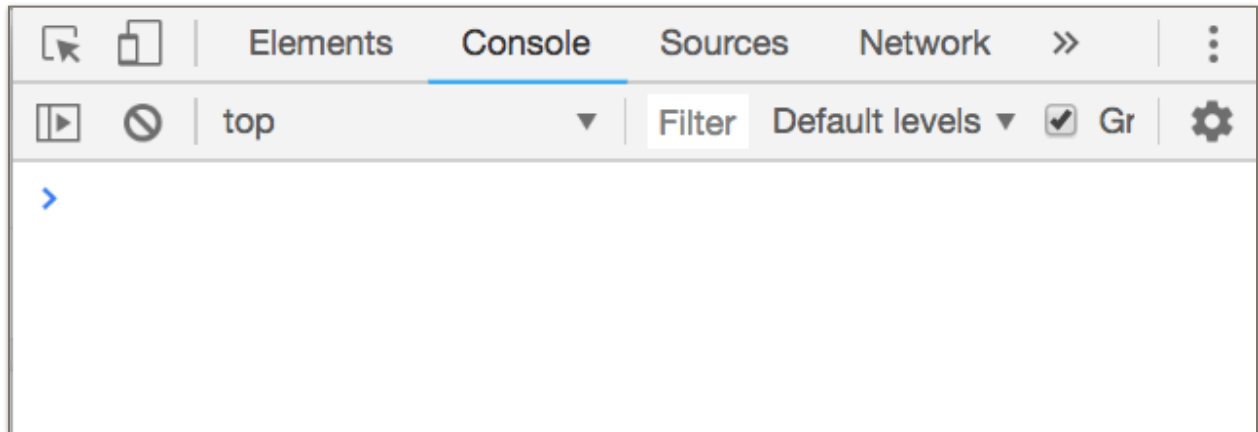


Javascript is **file** to write instructions to computer

In web - performs actions

# Javascript

Chrome Developer Tools > Console



To clear console: `clear()` or 

## Javascript Types

1. Number
2. String
3. Boolean
4. Undefined
5. Null
6. Symbol (new in ECMAScript 6)
7. Object

Use \ in front of quote to prevent error

```
> 'This isn't nice'
```

```
✖ Uncaught SyntaxError: Unexpected identifier VM1467:1
```

```
> 'This isn\'t nice'
```

```
< "This isn't nice"
```

```
> "She said "Hi"."
```

```
✖ Uncaught SyntaxError: Unexpected identifier VM1531:1
```

```
> "She said \"Hi\"."
```

```
< "She said "Hi"."
```

add strings together to concatenate

```
> "steph" + "bot"
```

```
< "stephbot"
```

string - string = NaN

```
> "hello" - "bye"
```

```
< NaN
```

string + number = string

```
> 10 + "34"
```

```
< "1034"
```

number - string = number

```
> 10 - "3"
```

```
< 7
```

## Template Strings (ES5/ES6 only)

- Skip escape character for quotes
- Denote variables within strings more conveniently
- Dynamic variables allowed

Normally, must be careful using quotes within a string...  
denote that they are okay using the JavaScript escape character,  
backwards backslash \

```
const name = "Sally";  
const age = 34;  
const pet = "horse";  
  
const greeting = "Hello " + name + " you\'re pet is a " + pet + "!"
```

```
> greeting  
◀ "Hello Sally you're pet is a horse!"
```

Use back ticks ` ` (above tab key) so that we can use double and single quotes within a string while skipping the escape character, and denote our variables more conveniently (can be dynamic!)

```
> const name = "Sally";  
const age = 34;  
const pet = "horse";  
  
const greetingBest = `Hello ${name}, aren't you ${age-10}? You have  
a ${pet}`
```

```
> greetingBest  
◀ "Hello Sally, aren't you 24? You have a horse"
```

## Advanced JavaScript Types: *Symbol* (ES5/ES6 only)

Symbols create a completely unique type, so that you can make sure there's never going to be any conflict.

```
> let sym1 = Symbol();  
    let sym2 = Symbol('foo');  
    let sym3 = Symbol('foo');
```

```
> sym2
```

```
< Symbol(foo)
```

```
> sym3
```

```
< Symbol(foo)
```

```
> sym2===sym3
```

```
< false
```

The symbol value is used as an identifier mostly for object properties... because sometimes you don't want object properties (if you have thousands of them) to collide and be the same ones because then they'll get bugs.

## Javascript Comparisons (Boolean)

!== not equal to

<=

=== equal to

>

>=

<

```
> true
< true    boolean
```

```
> false
< false   boolean
```

```
> 3 > 2
< true
```

```
> 5 > 10
< false
```

```
> 5 >= 5
< true
```

```
> 5 <= 5
< true
```

```
> 3 = 3    <<<<    = is for var assignment
```

✖ Uncaught ReferenceError: Invalid left-hand side in assignment VM841:1

Use === to check if they are equal, = gives error

```
> 3 === 3
< true    checks if equal to
```

```
> 3 !== 3
< false
> 4 !== 5
< true
```



## Module Operator % returns remainder (division)

```
> 12 % 6                                5 % 10 // 5
< 0
> 12 % 5
< 2
```

— exercise1.txt —

// Guess what answers you would get if you ran this in the  
// Javascript Console in Google Chrome. Once you have an  
answer to the questions then  
// check them by copying them and running it in the console  
yourself line by line

//Evaluate the below:

5 + "34"

5 - "4"

10 % 5

5 % 10

"Java" + "Script"

" " + " "

" " + 0

true + true

true + false

false + true

false - true

3 - 4

"Bob" - "bill"

//Evaluate the below comparisons:

5 >= 1

0 === 1

4 <= 1

1 != 1

"A" > "B"

"B" < "C"

"a" > "A"

"b" < "A"

true === false

true != true

// Make the string: "Hi There! It's "sunny" out" by using the + sign:

expression - code that produces a value  
\*\*\*\* needs to have a semicolon at the end \*\*\*\*

## Exercise 1 solutions

```
> 5 + "34"
```

```
< "534"
```

```
> 5 - "4"
```

```
< 1
```

```
> 10 % 5
```

```
< 0
```

```
> 5 % 10
```

```
< 5
```

```
> "Java" + "Script"
```

```
< "JavaScript"
```

```
> " " + " "
```

```
< " "
```

```
> " " + 0
```

```
< " 0"
```

```
> true + true
```

```
< 2
```

converts boolean to a number?

```
> true + false
```

true = 1

```
< 1
```

false = 0

```
> false + true
```

```
< 1
```

```
> false - true
```

```
< -1
```

```
> 3 - 4
```

```
< -1
```

```
> "Bob" - "bill"
```


```
< NaN
```

```
> 5 >= 1
< true
> 0 === 1
< false
> 4 <= 1
< false
> 1 != 1
< false
```

```
> "A" > "B"
< false
> "B" < "C"
< true
> "a" > "A"
< true
> "b" < "A"
< false
```

```
> true === false
< false
> true != true
< false
```

```
> true === false
< false
> true != true
< false
```



So how does this work....?

> "A" > "B"	"A" < "B" < "C"
< false	
> "A" < "B"	
< true	
> "B" < "C"	
< true	
> "B" > "C"	
< false	
> "a" > "A"	"A" < "a"
< true	
> "a" < "A"	
< false	
> "b" > "B"	
< true	
> "b" < "B"	
< false	
> "b" < "A"	
< false	
> "b" > "A"	
< true	

" " < "A" < "B" < "C" < "D" < ... < "Z" < "a" < "b" < ...

ABCD....Zabcd....z  
smallest^ ^largest