# JavaScript Quick Reference

Intermediate Level

Alexandra Babis
v0.3.2

## variables

```
let height = 5.4;
const FPS = 30;
```

```
var size = 50;
```

*similar to let but weird.*

## operators

```
 2 + 2  → 4          5 > 1   → true
 6 - 3  → 3          5 < 1   → false
 5 * 4  → 20         5 >= 5  → true
 3 / 2  → 1.5        5 <= 5  → true
 9 % 4  → 1         '4'== 4  → true
2 ** 3  → 8          5 != '5' → false
4 === 4 → true       5 !== '5' → true
```

*== is only recommended when comparing to null*

```
true && false → false      true && 8  → 8
true || false → true      false || 8  → 8
   0 ?? 20    → 0          null ?? 8  → 8
```

*?? is useful for assigning default values*

*Math.floor is needed for integer arithmetic*

```
const eggs = 29;
const cartons = Math.floor(eggs / 12);
const leftover = eggs % 12;
```

## functions

*function declaration*

```
function showResult(text) {
  document.body.innerText = text;
}
```

*arrow function*

```
const showResult = (text) => {
  document.body.innerText = text;
}
```

```
function getCirc(r) {
  return 2 * Math.PI * r;
}
const getCirc = (r) => 2 * Math.PI * r;
```

*Arrow functions make concise callbacks.*

```
setTimeout(() => alert('1s later'), 1000);
```

## strings

```
let name = 'Alan Turing';
name[0];                        // 'A'
name[1];                        // 'l'
name[name.length - 1];          // 'g'
```

```
'5' + '5'                       // '55'
```

*A leading plus converts a string into a number*

```
const userInput = '19';         // '19'
const age = +userInput;         //  19
const nextYear = age + 1;       //  20
```

*Template strings can include variables concisely*

```
const m = `Age next year: ${nextYear}`;
```

## objects

```
const detective = {
  firstName: 'Sherlock',
  lastName: 'Holmes',
};
const brother = {
  ...detective,
  firstName: 'Mycroft',
};
detective.firstName; // 'Sherlock'
brother.lastName;    // 'Holmes'
```

## loops

```
let langs = ['C', 'C++', 'C#'];
for (const lang of langs)
  console.log(lang);
```

```
for (let x = 1; x <= 10; x++) {
  const y = i * 3;
  console.log(`${x} times 3 is ${y}`);
}
```

```
while (current = queue.getNext()) {
  console.log(current.value);
}
```

## arrays *(lists)*

```
const colors = ['red', 'brown', 'blue'];
colors[0];                    // 'red'
colors[1];                    // 'brown'
colors.length                 // 3
colors[colors.length - 1]     // 'blue'
colors.at(-1)                 // 'blue'
```

## modifying arrays

| | |
|---|---|
| Add to the end | `list.push('banana')` |
| Add to the start | `list.unshift('banana')` |
| Remove from end | `list.pop()` |
| Remove from start | `list.shift()` |
| Remove at index | `list.splice(index, 1)` |
| Insert at index | `list.splice(index, 0, x)` |

## searching arrays

```
const temps = [63, 70, 81, 73, 63, 72];

Math.min(...temps)            → 63
Math.max(...temps)            → 81
temps.includes(70)            → true
temps.indexOf(63)             → 0
temps.lastIndexOf(63)         → 4
temps.find(t => t > 70)       → 81
temps.find(t => t > 90)       → undefined
temps.findIndex(t => t > 70)  → 2
temps.findIndex(t => t > 90)  → -1
```

## copying arrays

```
const list1 = [1, 2, 3];
const list2 = [4, 5, 6];

const list3 = [...list1];  // [1, 2, 3];
list1 === list3;           // false

// [1, 2, 3, 4, 5, 6]
const list4 = [...list1, ...list2];
```

## string functions

```
'Banana'.substring(3)     → 'ana'
'Banana'.indexOf('n')     → 2
'Banana'.includes('ana')  → true
'Banana'.toUpperCase()    → 'BANANA'
'Banana'.toLowerCase()    → 'banana'
'$85.63'.padStart(10)     → '    $86.63'
'Potato'.padEnd(10, '.')  → 'Potato....'
'Hi    '.trim()           → 'Hi'
'A-Z'.split('-')          → ['A', 'Z']
```

## data processing

```
const myList = [4, 0, 1, 3];
const sum = (a, b) => a + b;

myList.filter(x => x > 2) → [4, 3]
myList.map(x => x * 2)    → [8, 0, 2, 6]
myList.reduce(sum)        → 8
myList.forEach((value) => {
  console.log(value);
});
myList.every(x => x > 5) → false
myList.some(x => x > 2)  → true
myList.join(':')         → '4:0:1:3'
myList.fill(0)           → (fills list with 0s)
```

## destructuring

```
const name = 'Samuel L Jackson';
const parts = name.split(' ');

const [first, middle, last] = parts;

`${last}, ${first}` //  Jackson, Samuel
```

```
const { age }    = detective;  // 37
const { length } = colors;     // 3
```

```
document.body.onclick = ({ target }) => {
  console.log(target);
}
```