JavaScript ES6 Cheat Sheet

@codinatute

```
Object property assignment
                                                                                 Object function assignment
Arrow function
const sum = (a, b) \Rightarrow a + b
                                                                                 const obj = {
                                          const a = 2
                                                                                  a: 5,
console.log(sum(2, 6)) // prints 8
                                          const b = 5
                                                                                  b() {
                                          const obj = { a, b }
Default parameters
                                                                                  console.log('b')
                                          // Before es6:
function print(a = 5) {
                                                                                  }
                                          // obj = { a: a, b: b }
 console.log(a)
                                          console.log(obj)
                                                                                 obj.b() // prints "b"
                                          // prints { a: 2, b: 5 }
print() // prints 5
print(22) // prints 22
                                           Object.assign()
                                                                                 Object.entries()
                                          const obj1 = \{a: 1\}
Let Scope
                                                                                 const obj = {
                                          const obj2 = \{ b: 2 \}
                                                                                  firstName: 'FirstName',
let a = 3
                                          const obj3 = Object.assign({}),
                                                                                  lastName: 'LastName',
if (true) {
                                               obj1, obj2)
                                                                                  age: 24,
let a = 5
                                          console.log(obj3)
                                                                                  country: 'India',
 console.log(a) // prints 5
                                           // { a: 1, b: 2 }
                                                                                 };
                                                                                 const entries = Object.entries(obj);
console.log(a) // prints 3
                                                                                 /* returns an array of [key, value]
                                           Promises with finally
                                                                                  pairs of the object passed */
                                           promise
// can be assigned only once
                                                                                 console.log(entries);
                                            .then((result) => { ··· })
const a = 55
                                                                                 /* prints
                                            .catch((error) => { ··· })
a = 44 // throws an error
                                            .finally(() \Rightarrow { /* logic}
                                                                                  ['firstName', 'FirstName'],
                                           independent of success/error */ })
Multiline string
                                                                                  ['lastName', 'LastName'],
                                           /* The handler is called when the
console.log(`
                                                                                  ['age', 24],
                                           promise is fulfilled or rejected.*/
This is a
                                                                                  ['country', 'India']
multiline string
                                                                                  ]; */
`)
Template strings
                                                Spread operator
const name = 'World'
                                                const a = {
const message = `Hello ${name}`
                                                 firstName: "FirstName",
console.log(message)
                                                 lastName: "LastName1",
// prints "Hello World"
                                                const b = {
   ponent operator
const byte = 2 ** 8
                                                 ...a,
// Same as: Math.pow(2, 8)
                                                 lastName: "LastName2",
                                                 canSing: true,
Spread operator
                                                }
const a = [1, 2]
                                                console.log(a)
const b = [3, 4]
                                                //{firstName: "FirstName", lastName: "LastName1"}
const c = [ ...a, ...b ]
console log(c) // [1, 2, 3, 4]
                                                console.log(b)
                                                /* {firstName: "FirstName", lastName: "LastName2",
String includes()
                                                canSing: true} */
console.log('apple'.includes('pl'))
                                                /* great for modifying objects without side
// prints true
console.log('apple'.includes('tt'))
                                                effects/affecting the original */
// prints false
String startsWith()
                                                Destructuring Nested Objects
console.log('apple'.startsWith('ap'))
                                                const Person = {
//prints true
                                                 name: "Harry Potter",
console.log('apple'.startsWith('bb'))
//prints false
                                                 age: 29,
                                                 sex: "male",
String repeat()
                                                 materialStatus: "single",
console.log('ab'.repeat(3))
                                                 address: {
//prints "ababab"
                                                 country: "USA",
                                                 state: "Nevada",
Destructuring array
                                                 city: "Carson City",
let [a, b] = [3, 7];
                                                 pinCode: "500014",
console.log(a); // 3
                                                 },
console.log(b); // 7
                                                };
Destructuring object
                                                const { address : { state, pinCode }, name } = Person;
let obj = {
                                                console.log(name, state, pinCode)
a: 55,
                                                // Harry Potter Nevada 500014
b: 44
                                                console.log(city) // ReferenceError
};
let { a, b } = obj;
```

console.log(a); // 55
console.log(b); // 44

JavaScript ES6 Cheat Sheet

@codinatute

Arrow function

```
const sum = (a, b) \Rightarrow a + b
console.log(sum(2, 6)) // prints 8
```

Default parameters

```
function print(a = 5) {
  console.log(a)
}
print() // prints 5
print(22) // prints 22
```

Let Scope

```
let a = 3
if (true) {
  let a = 5
  console.log(a) // prints 5
}
console.log(a) // prints 3
```

Const

```
// can be assigned only once const a = 55 a = 44 // throws an error
```

Multiline string

```
console.log(`
  This is a
  multiline string
`)
```

Template strings

```
const name = 'World'
const message = `Hello ${name}`
console.log(message)
// prints "Hello World"
```

Exponent operator

```
const byte = 2 ** 8
// Same as: Math.pow(2, 8)
```

Spread operator

```
const a = [ 1, 2 ]
const b = [ 3, 4 ]
const c = [ ...a, ...b ]
console.log(c) // [1, 2, 3, 4]
```

String includes()

```
console.log('apple'.includes('pl'))
// prints true
console.log('apple'.includes('tt'))
// prints false
```

String startsWith()

```
console.log('apple'.startsWith('ap'))
//prints true
console.log('apple'.startsWith('bb'))
//prints false
```

String repeat()

```
console.log('ab'.repeat(3))
//prints "ababab"
```

Destructuring array

```
let [a, b] = [3, 7];
console.log(a); // 3
console.log(b); // 7
```

Destructuring object

```
let obj = {
  a: 55,
  b: 44
};
let { a, b } = obj;
console.log(a); // 55
console.log(b); // 44
```

Object property assignment

```
const a = 2
const b = 5
const obj = { a, b }
// Before es6:
// obj = { a: a, b: b }
console.log(obj)
// prints { a: 2, b: 5 }
```

Object.assign()

```
const obj1 = { a: 1 }
const obj2 = { b: 2 }
const obj3 = Object.assign({},
    obj1, obj2)
console.log(obj3)
// { a: 1, b: 2 }
```

Promises with finally

```
promise
  .then((result) => { ··· })
  .catch((error) => { ··· })
  .finally(() => { /* logic
  independent of success/error */ })
/* The handler is called when the
promise is fulfilled or rejected.*/
```

Object function assignment

```
const obj = {
  a: 5,
  b() {
  console.log('b')
  }
}
obj.b() // prints "b"
```

Object.entries()

```
const obj = {
 firstName: 'FirstName',
 lastName: 'LastName',
age: 24,
country: 'India',
};
const entries = Object.entries(obj);
/* returns an array of [key, value]
 pairs of the object passed */
console.log(entries);
/* prints
 ['firstName', 'FirstName'],
 ['lastName', 'LastName'],
 ['age', 24],
 ['country', 'India']
 ]; */
```

Spread operator

```
const a = {
  firstName: "FirstName",
  lastName: "LastName1",
}
const b = {
    ...a,
  lastName: "LastName2",
    canSing: true,
}
console.log(a)
//{firstName: "FirstName", lastName: "LastName1"}
console.log(b)
/* {firstName: "FirstName", lastName: "LastName2",
  canSing: true} */
/* great for modifying objects without side
effects/affecting the original */
```

Destructuring Nested Objects

```
const Person = {
name: "Harry Potter",
age: 29,
 sex: "male",
 materialStatus: "single",
address: {
country: "USA",
 state: "Nevada",
 city: "Carson City",
pinCode: "500014",
},
};
const { address : { state, pinCode }, name } = Person;
console.log(name, state, pinCode)
// Harry Potter Nevada 500014
console.log(city) // ReferenceError
```