Day 13 - Console Object Methods

Absolute beginners usually don’t know which to use: console.log(), document.write() or document.getElementById().

We use console object methods to show output to the browser console and we use document.write to show output on the browser document(view port). Both are used only for testing and debugging purposes. Console method is the most popular testing and debugging tool on the browser. document.getElementById() is used when we want to interact with the DOM using JavaScript.

console.log()

Shows output to the browser console. We can substitute values and also style the logging output using %c.

Showing output on browser console:

```
console.log('30 Days of JavaScript')
// 30 Days of JavaScript
```

Substitution:

```
console.log('%d %s of JavaScript', 30, 'Days')
// 30 Days of JavaScript
```

CSS

We can style logging message with CSS.

```
console.log('%c30 Days of JavaScript', 'color:green') // log output is green
console.log(
  '%c30 Days%c %cOf%c %cJavaScript%c',
  'color:green',
  '',
  'color:red',
  '',
  'color:yellow'
) // log output green red and yellow text
```

console.warn()

We use console.warn() to give warning on browser. For instance, we can warn deprecation of version of a package or bad practices.

```
console.warn('This is a warning')
console.warn(
  'You are using React. Do not touch the DOM. Virtual DOM will take care of handling the DOM!'
)
console.warn('Warning is different from error')
```

console.error()

Shows an error message.

```
console.error('This is an error message')
console.error('We all make mistakes')
```

console.table()

Displays data as a table on the console. Displays tabular data as a table. Method takes one required argument data, which must be an array or an object, and one additional optional columns parameter.

The code below displays a table with two columns. An index columns displays the index and the value column displays the names.

```
const meals = ['Taco', 'Sandwich',' Salad']
console.table(meals)

const user = {
    fName: 'Chris',
    lName: 'Guerrero',
    age: 24,
    state: 'Arizona',
}
console.table(user)
```

More examples:

```
const states = [
    ['Alaska', 'Anchorage'],
    ['Washington', 'Seattle'],
    ['New York', 'Brooklyn']
]
console.table(states)

┌─────────┬──────────────┬─────────────┐
│ (index) │      0       │      1      │
├─────────┼──────────────┼─────────────┤
│    0    │   'Alaska'   │ 'Anchorage' │
│    1    │ 'Washington' │  'Seattle'  │
│    2    │  'New York'  │ 'Brooklyn'  │
└─────────┴──────────────┴─────────────┘

const states = [
    {state: 'Alaska', city: 'Anchorage'},
    {state: 'California', city: 'Los Angeles'},
    {state: 'New York', city: 'Brooklyn'},
    {state: 'Washington', city: 'Seattle'}
]
console.table(states)

┌─────────┬──────────────┬───────────────┐
│ (index) │    state     │     city      │
├─────────┼──────────────┼───────────────┤
│    0    │   'Alaska'   │  'Anchorage'  │
│    1    │ 'California' │ 'Los Angeles' │
│    2    │  'New York'  │  'Brooklyn'   │
│    3    │ 'Washington' │   'Seattle'   │
└─────────┴──────────────┴───────────────┘
```

console.time()

Starts a timer to track how long an operation takes. Each timer gets a unique name, and you may have up to 10,000 timers running on a given page. When console.timeEnd() is called with the same timer name, the browser outputs the time, in milliseconds, that elapsed since the timer was started.

```
const countries = [
  ['Finland', 'Helsinki'],
  ['Sweden', 'Stockholm'],
  ['Norway', 'Oslo']
]
```

```
console.time('Regular for loop')
for (let i = 0; i < countries.length; i++) {
  console.log(countries[i][0], countries[i][1])
}
console.timeEnd('Regular for loop')

console.time('for of loop')
for (const [name, city] of countries) {
  console.log(name, city)
}
console.timeEnd('for of loop')

console.time('forEach loop')
countries.forEach(([name, city]) => {
  console.log(name, city)
})
console.timeEnd('forEach loop')

// log
Finland Helsinki
Sweden Stockholm
Norway Oslo
Regular for loop: 3.754ms
Finland Helsinki
Sweden Stockholm
Norway Oslo
for of loop: 0.107ms
Finland Helsinki
Sweden Stockholm
Norway Oslo
forEach loop: 0.725ms
```

console.info

Displays information message on browser console.

```
console.info('30 Days Of JavaScript challenge is trending on Github')
console.info('30 Days Of fullStack challenge might be released')
console.info('30 Days Of HTML and CSS challenge might be released')
```

console.assert()

Writes an error message to the console if assertion is false. If assertion is true, nothing happens. First parameter is an assertion expression. If expression is false, and assertion failed error message will be displayed.

```
console.assert(4 > 3, '4 is greater than 3') // no result
console.assert(3 > 4, '3 is not greater than 4') // Assertion failed: 3 is not greater than 4
```

console.group()

Can help to group different log groups.

```
const names = ['Asabeneh', 'Brook', 'David', 'John']
const countries = [
  ['Finland', 'Helsinki'],
  ['Sweden', 'Stockholm'],
  ['Norway', 'Oslo']
]
const user = {
  name: 'Asabeneh',
  title: 'Programmer',
  country: 'Finland',
  city: 'Helsinki',
  age: 250
}
const users = [
  {
    name: 'Asabeneh',
    title: 'Programmer',
    country: 'Finland',
    city: 'Helsinki',
    age: 250
  },
  {
    name: 'Eyob',
    title: 'Teacher',
    country: 'Sweden',
    city: 'London',
    age: 25
  },
  {
    name: 'Asab',
    title: 'Instructor',
    country: 'Norway',
    city: 'Oslo',
    age: 22
  },
  {
    name: 'Matias',
    title: 'Developer',
    country: 'Denmark',
    city: 'Copenhagen',
    age: 28
  }
]
```

```
console.group('Names')
console.log(names)
console.groupEnd()

console.group('Countries')
console.log(countries)
console.groupEnd()

console.group('Users')
console.log(user)
console.log(users)
console.groupEnd()
```

console.count()

Prints number of times the console.count() method is called. Takes a string label parameter. Very helpful for counting the number of times a function is called.

```
const func = () => {
  console.count('Function has been called')
}
func()
func()
func()

// log:
Function has been called: 1
Function has been called: 2
Function has been called: 3
```

console.clear()

Cleans the browser console