Frontender[1.0] JavaScript - Numbers, Числа, примитивы как объекты, методы чисел

YouTube	https://youtu.be/48CSqrAg6ro
Telegram	https://t.me/Dmitry_Kolotilshikov
Github	https://github.com/DmitryKolotilshikov/
Boosty	https://boosty.to/dmitry_ko
# Номер урока	10

💡 Примитив как объект

Вот парадокс, с которым столкнулся создатель JavaScript:

- Есть много всего, что хотелось бы сделать с примитивами, такими как строка или число. Было бы замечательно, если бы мы могли обращаться к ним при помощи методов.
- Примитивы должны быть лёгкими и быстрыми насколько это возможно.

Выбранное решение, хотя выглядит оно немного неуклюже:

1. Примитивы остаются примитивами. Одно значение, как и хотелось.
2. Язык позволяет осуществлять доступ к методам и свойствам строк, чисел, булевых значений и символов.
3. Чтобы это работало, при таком доступе создаётся специальный «объект-обёртка», который предоставляет нужную функциональность, а после удаляется.

Каждый примитив имеет свой собственный «объект-обёртку», которые называются: `String`, `Number`, `Boolean`, `Symbol` и `BigInt`. Таким образом, они имеют разный набор методов.

- Все примитивы, кроме `null` и `undefined`, предоставляют множество полезных методов. Мы познакомимся с ними поближе в следующих главах.
- Формально эти методы работают с помощью временных объектов, но движки JavaScript внутренне очень хорошо оптимизируют этот процесс, так что их вызов не требует много ресурсов.

> **Конструкторы `String/Number/Boolean` предназначены только для внутреннего пользования**
>
> Некоторые языки, такие как Java, позволяют явное создание «объектов-обёрток» для примитивов при помощи такого синтаксиса как `new Number(1)` или `new Boolean(false)`.
>
> В JavaScript, это тоже возможно по историческим причинам, но очень **не рекомендуется**. В некоторых местах последствия могут быть катастрофическими.
>
> Например:
>
> ```
> alert( typeof 0 ); // "число"
>
> alert( typeof new Number(0) ); // "object"!
> ```
>
> Объекты в `if` всегда дают `true`, так что в нижеприведённом примере будет показан `alert`:
>
> ```
> let zero = new Number(0);
>
> if (zero) {
>   // zero возвращает "true", так как является объектом
>   alert( "zero имеет «истинное» значение?!?" );
> }
> ```
>
> С другой стороны, использование функций `String/Number/Boolean` без оператора `new` – вполне разумно и полезно. Они превращают значение в соответствующий примитивный тип: в строку, в число, в булевый тип.
>
> К примеру, следующее вполне допустимо:
>
> ```
> let num = Number("123"); // превращает строку в число
> ```

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <script src="script.js"></script>
    <title>Numbers</title>
</head>
<body>
</body>
</html>
```

```
// Numbers (числа)

let x = 10;
x = Number(10); // 10
x = new Number(10); // object
x.valueOf() // 10
```

```
// console.log(x);

// -------------- Методы --------------

let num = 20;

// num = num.toFixed(2); // 20.00
// num = num.toFixed(3); // 20.000
// num = num.toFixed(); // 20

// num = num.toPrecision(3); // 20.0

// num = num.toString();
// console.log(typeof num);

// num = num.toString().length; // 2

// console.log(num);
console.log(num.valueOf());

const maxNumber = Number.MAX_VALUE;
console.log(maxNumber);

const minNumber = Number.MIN_VALUE;
console.log(minNumber);

/*
Примитив как объект
Каждый примитив имеет свой собственный «объект-обёртку»,
которые называются: String, Number, Boolean, Symbol и BigInt.

Таким образом, они имеют разный набор методов.
*/
```