Number and String

План

- задание числа
- проверка на число
- округления чисел
- объект Math
- случайные числа
- объект Date
- задание сроки
- длина
- доступ к символам
- поиск позиции подстроки
- копирование подстроки

Запись числа

```
315 - целое

3.15 - дробное

3.15e2 - с плавающей точкой 3.15 * 100 = 315

3.15e-2 - если отрицательный степень 3.15 * 0.01 = 0.0315

0xFF - шестнадцатеричный запись // 255
```

Infinity

- Математическая бесконечность

```
5155/0          // Infinity

-8458/0         // -Infinity

0/0             // NaN

Infinity>3155644984646     // true
```

NaN

- Not a Number (не число)
- Если невозможно выполнить действие то результат равен NaN
- NaN не ровен никакому значению включая себя
- NaN == NaN // false

```
10/‘10px’     // NaN

10+NaN        // NaN
```

Приведение к числу

- parseInt () - приводит строки к числу, если она начинается на число
- parseFloat () - приводит к числу символ за символом, пока это возможно

```
parseInt(‘25mm’);          // 25

parseFloat(‘25.2mm’);      // 25.2

parseFloat(‘25.2.3’);      // 25.2

parseFloat(true);          // NaN
```

object Number

- число можно определить как объект через new Number ();
- typeof возвращает объект
- можно использовать для приведения в число
- Number (false); // 0
- Number ('100a') // NaN

Округления чисел

- Math.floor - округляет вниз
- Math.ceil - округляет вверх
- Math.round - округляет до ближайшего целого
- Math.trunc - отвергает дробную частичку

	Math.floor	Math.ceil	Math.round	Math.trunc
3.1	3	4	3	3
3.6	3	4	4	3
-1.1	-2	-1	-1	-1
-1.6	-2	-1	-2	-1

Округление до заданной точности

```
var n = 26.1234;

n.toFixed(1);         // 26.1

n.toFixed(5);         // 26.12340 добавляет нули в конце

Math.round(n * 100) / 100;  // 26.1234 - 2612.34 - 2612 - 26.12
```

object Math

- объект с математическими функциями

https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Global_Objects/Math

Случайные числа(random)

Math.random () - возвращает случайное число между 0 и 1

Math.floor (Math.random () * 10) + 1; - случайное число от 1 до 10

Общая формула случайного числа:

```
Math.floor(Math.random() * (max - min +  1) ) + min;
```

Практика

- функция принимает пределы случайного числа (min, max) и вывести в консоль четное или нечетное оно

Date

- для работы с датой и временем в JavaScript используются объекты Date объект с текущей датой

```
let now = new Date();
alert( now );
```

Объект Date, значение которого равно количеству миллисекунд (1/1000 секунды), прошедших с 1 января 1970 GMT + 0.

https://learn.javascript.ru/datetime

Задание даты

```
new Date(year, month, date, hours, minutes, seconds, ms)
```

```
new Date(2011, 0, 1, 0, 0, 0, 0); // // 1 січень 2011, 00:00:00
```

Получение даты

getFullYear () Получить год как четырехзначное число (yyyy)

getMonth () Получить месяц как число (0-11)

getDate () Получить день как число (1-31)

getHours () Получить час (0-23)

getMinutes () Получить минуту (0-59)

getSeconds () Получить секунду (0-59)

getMilliseconds () Получить миллисекунду (0-999)

getTime () Получите время (миллисекунды с 1 января 1970)

getDay () Получить рабочий день как число (0-6)

https://www.w3schools.com/js/js_date_methods.asp

Задать даты

setDate () Установить день как число (1-31)

setFullYear () Установить год (необязательно месяц и день)

setHours () Установить час (0-23)

setMilliseconds () Установить миллисекунды (0-999)

setMinutes () Установить минуты (0-59)

setMonth () Установить месяц (0-11)

setSeconds () Установить секунды (0-59)

setTime () Установить время (миллисекунды с 1 января 1970)

https://www.w3schools.com/js/js_date_methods_set.asp

Задать строки

Варианты, как задать строку

```
let str = "Привет";

let str2 = 'одинарные кавычки';

let phrase = `Обратные кавычки`;
```

Вставка переменной в строку

```
let price = 100;

let strPrice = 'Ваша ціна' + price + 'грн.' ;

let strPrice = `Ваша ціна ${price} грн.`;
```

В первом варианте конкатинируем строку с числом, в случае с обратными кавычками вставляем переменную в любое место строки из $ {...}

Экранирование

- если нужно вставить в строку спецсимвол то используем экранирование через \

  I \ 'm a JavaScript programmer';

  "I'm a JavaScript \" programmer \ "";

object String

- сроку можно определить как объект через new String ();
- typeof возвращает объект
- можно использовать для приведения в сроки
- String (10); // '10'

Длина строки length

```
var str = ‘I am string’;

str.length;      // 11
```

Доступ к символам

- чтобы получить элемент используется charAt (позиция)
- через квадратные скобки [позиция]

```
var str = ‘Some string’;
str.charAt(0);        // ‘S’
str[0];               // ‘S’
"" .charAt (0);       // пустая срока
"" [0];               // undefined
```

- str.charCodeAt(символ) - вернет unicode символа

Строки неизменны

Строки в JavaScript нельзя изменить. Нельзя взять символ посередине и заменить его. Как только строка создана - она такая навсегда.

```
let str = 'Hi';

str[0] = 'h'; // error
```

Изменение регистра

- str.toUpperCase () - делает в строке все большие буквы
- str.toLowerCase () - делает в строке все прописные буквы

```
var str = "stringify";

str.toUpperCase(); // "STRINGIFY",
```

Позиция подстроки в строке indexOf

- вернет позицию на которой находится подстрока
- если не найдено то вернет -1
- второй параметр указывает на позиция по которой начать поиск,
- str.indexOf ('sds ", position)

```
var str = "Please locate where locate occurs!";

var pos = str.indexOf("locate");
```

- есть аналогичный метод lastIndexOf который ищет с конца строки

includes()

- проверяет, содержит ли строка заданную подстроку, и возвращает, соответственно true или false

```
var str = "Please locate where locate occurs!";

var pos = str.includes("locate");                    // true
```

startsWith endsWith

Методы str.startsWith и str.endsWith проверяют, соответственно, начинается ли и заканчивается строка определенной строкой:

```
alert( "Widget".startsWith("Wid") ); // true, "Wid" — начало

alert( "Widget".endsWith("get") ); // true, "get" — окончания
```

trim()

- удаляет пробелы с двух сторон строки

```
var str = "    Hello World!        ";

str.trim(); // “Hello World!”
```

Копирование подстроки

- `substring(start [, end])` возвращает подстроку с позиции start до исключая end

```
var str= "stringify";

str.substring(0,1); // "s",
```

- `substr(start [, length])` возвращает подстроку с позиции start, второй параметр количество символов

```
str.substr(1,4); // "trin",
```

- `slice(start [, end])` возвращает подстроку с позиции start до исключая end

```
var str = "stringify";

str.slice(0,1); // "s",
```

- отличие в том, что slice может работать с отрицательными значениями которые отсчитываются от конца строки

Сравнение строк

- строки сравниваются в алфавитном порядке и посимвольно
- сравнивается код в кодировке unicode
- строчные буквы больше крупных

```
alert( 'a' > 'Z' ); // true
```

Позиция подстроки в строке search ()

- вернет позицию на которой находится подстрока
- если не найдено то вернет -1
- поддерживает регулярные выражения

str.search(‘substring’);

replace()

- Меняет подстроку на подстроку

```
str = "Please visit Microsoft!";

var n = str.replace("Microsoft", "W3Schools"); // Please visit W3Schools!
```

Практика

- есть строка "lorem ipsum is simply dummy", сделать новую строку, чтобы первое слово было с большой буквы
- функция принимает две строки и возвращает большую из них

Регулярные выражения

это шаблоны используются для сопоставления последовательностей символов с шаблоном. Эти шаблоны используются в методах exec и test объекта RegExp, а также match, replace, search, и split объекта String.

Регулярные выражения в JS

- В JavaScript регулярные выражения реализованы отдельным объектом RegExp и интегрированы в методы строк

```
var regexp = new RegExp("шаблон", "флаги");
```

краткая запись (чаще используется)

```
var regexp = /шаблон/; // без флагов
var regexp = /шаблон/gmi; //  з флагами gmi
```

Флаги

i

Если этот флаг является, то "регулярка" ищет независимо от регистра, то есть не различает между A и a.

g

Если этот флаг является, то "регулярка" ищет все совпадения, иначе - только первое.

m

Многострочный режим.

Методы

- search () - поиск в строке по указанному регулярным выражением, возвращает индекс на котором я совпадение и только первое
- match () - ищет и возвращает (если есть) соответствия строки с указанным регулярного выражения.
- test () - выполняет поиск на совпадение между регулярным выражением и заданным строкой. Возвращает true или false.
- replace () - возвращает новую строку с некоторыми или всеми сравнениями с шаблоном, замененными на заменитель

https://learn.javascript.ru/regexp-methods

пример

- найти количество вхождения подстроки в строку

```
var str = "Я люблю JavaScript!"; // будем искать в этой строке

alert (str.search (new RegExp('ЛЮ'))); // -1
alert (str.search (/ЛЮ/ i)); // 2
```

Классы символов

специальное обозначение, под которое подходит любой символ из определенного набора.

\ d - цифры.
\ D - не цифры.
\ s - пробельные символы переноса строки.
\ S - все, кроме \ s.
\ w - латиница, цифры, подчеркивания '_'.
\ W - все, кроме \ w.
'.' - точка обозначает любой символ, кроме переноса строки.
Если хочется поискать именно сочетание "\ d" или символ «точка», то его экранируют обратным слэшем, вот так: \.

https://learn.javascript.ru/regexp-character-classes

Примеры

- найти все числа в номере телефона

```
var str = "+7 (903) -123-45-67";
var reg = /\d/g;
alert (str.match(reg)); // массив всех совпадений: 7,9,0,3,1,2,3,4,5,6,7
```

Диапазоны

Квадратные скобки могут также содержать диапазоны символов.
Например, [a-zA-Z] - произвольный символ от a до z, [а-я] - произвольный символ от а до я, [0-5] - цифра от 0 до 5.

Квадратные скобки, начинающиеся со знака каретки: [^ ...] находят любой символ, кроме указанных
[^ 0-9] - любой символ, кроме цифры, тоже \ D.

```
alert( "width: 100px".match(/[0-9][0-9][0-9]px/g) ); // 100px
```

Ссылки

https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Global_Objects/String

https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Global_Objects/Number

https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Global_Objects/RegExp

https://developer.mozilla.org/ru/docs/Web/JavaScript/Guide/Regular_Expressions

РЕГУЛЯРКИ практика

Видео

https://www.youtube.com/watch?v=9hLkbhRs7jM

https://www.youtube.com/watch?v=CAXBO9jOXFA