Week 17

1. Для создания строк можно использовать как одинарные, так и двойные кавычки:

var carname = "Вольво XC60";

var carname = 'Вольво XC60';

Можно использовать кавычки внутри строки, пока они не будут соответствовать кавычкам вокруг строки:

var answer1 = "Её зовут 'Маша'";

var answer2 = 'Её зовут "Маша"';

Также можно использовать третий вид кавычек - обратные.

Если использовать обратные кавычки, то в такую строку мы сможем вставлять произвольные выражения, обернув их в `${…}`:

function sum(a, b) {

return a + b;

}

alert(`1 + 2 = ${sum(1, 2)}`); // 1 + 2 = 3

С такой конструкцией мы можем также делать персонализированные обращения:

function welcome(name) {

alert(`Привет, ${name}!`);

}

welcome('Алиса'); // Привет, Алиса!

Ещё одно преимущество обратных кавычек — они могут занимать более одной строки:

let guestList = `Гости:

\* Юля

\* Петя

\* Маша

`;

alert(guestList); // список гостей, состоящий из нескольких строк

2. Метод `indexOf()` возвращает индекс (позицию) в первое вхождение указанного текста в строку.

Метод `lastIndexOf()` возвращает индекс последнего вхождения указанного текста в строку:

Оба метода `indexof()` и `lastIndexOf()` возвращают значение `-1`, если текст не найден.

Похожим образом работает метод `search()` - он тоже ищет строки по заданному значению и возвращает позицию совпадения.

Различия в том, что метод `search()` может принимать гораздо более мощные значения поиска (регулярные выражения).

let text = "I'm learning JavaScript and now I can say, that I like to learn JavaScript";

console.log(text);

let place = text.indexOf('JavaScript');

console.log(place); //13

let lastPlace = text.lastIndexOf('JavaScript');

console.log(lastPlace); //64

let searchPlace = text.search('JavaScript');

console.log(searchPlace); //13

3. «символ перевода строки», который записывается как \n используется для создания многострочных строк с помощью одинарных и двойных кавычек.

let list = "List:\nApples\nGrape\nMelon\nOrange";

alert(list);

4.делаем первый символ заглавным

let str = 'настя';

let newStr = str[0].toUpperCase() + str.slice(1);

alert(newStr);

function changeFirst(str) {

if (!str) return str;

return str[0].toUpperCase() + str.slice(1);

}

alert( changeFirst("настя") ); // Настя

5. 24 января

let numm = new Date(2021,0,24,22,51); //0 это январь

console.log(numm);

6. Секунды до завтра. Как получить при помощи JS разницу между текущим временем и полночью нового дня - пока не поняла.

Вариант1: я сама задаю завтрашнюю дату, полночь (но здесь нет автоматизма и завтрашняя дата не будет сама вычисляться:

let dateTomorrow = new Date(2022,9,2,00,00);

console.log(dateTomorrow);

let now = new Date();

console.log(now);

let diffTomorrow = dateTomorrow - Date.now();

console.log(diffTomorrow/1000); //получили секунды

Вариант2: отсчитывается 24 часа от текущего времени(но это не очень корректно, итак понятно, что разница будет 24 часа):

let today = new Date();

let timeToday = new Date(today.getTime());

console.log(timeToday);

let tomorrow = new Date(today.getTime() + (24 \* 60 \* 60 \* 1000));

let dayTomorrow = tomorrow.getDate();

let monthTomorrow = tomorrow.getMonth() + 1; //в js месяц отсчитывается с нуля

let yearTomorrow = tomorrow.getFullYear();

let different = tomorrow-today;

console.log(different/1000); //86400 секунд, т.е. 24 часа

7. Цифры можем выделить при помощи регулярного выражения

let value = "120EUR";

console.log(value.replace(/[^0-9]/g,""));

8. Join, split  
  
Строку можно преобразовать в массив с методом `split()`:

fio = document.getElementById('fio').value;

arrayFio = fio.split(' ');

console.log(arrayFio);

Типичная задача для этой функции: разделить адрес на части (страна, город и пр.) по запятым или ФИО по пробелам.

Обратное действие (склеить из массива строку) - это функция `join`.

l

et arr = [ 1, 2, 3 ];

arr.join('+'); // "1+2+3"

arr.join();  // "1,2,3" если без параметров, то автоматически ставится запятая как разделитель

9. Math.round(параметр) – математическое округление, т.е. когда дробная часть числа больше или равна 0.5, то оно округляется в большую сторону, а иначе в меньшую сторону;

console.log(Math.round(34.3223)); //34

10. Случайное число от 1 до 100

console.log(Math.random()\*100); //до 100

11. Функция str.trim() — убирает пробелы в начале и конце строки.

let cleanFamilia = familia.trim(familia);

console.log(cleanFamilia);

12. Флаги  
Регулярные выражения могут иметь флаги, которые влияют на поиск.

В JavaScript их всего шесть:

i

С этим флагом поиск не зависит от регистра: нет разницы между A и a .

g

С этим флагом поиск ищет все совпадения, без него – только первое.

m

Многострочный режим.

s

Включает режим «dotall», при котором точка . может соответствовать символу перевода строки \n.

u

Включает полную поддержку Юникода. Флаг разрешает корректную обработку суррогатных пар.

y

Режим поиска на конкретной позиции в тексте.