JavaScript на одном листе

Переменные

Ключевые слова: **const**, **let**, **var***(устар)*	Переменная может содержать любые типы данных. Допускается изменение типа. **camelStyleForMyConstName**

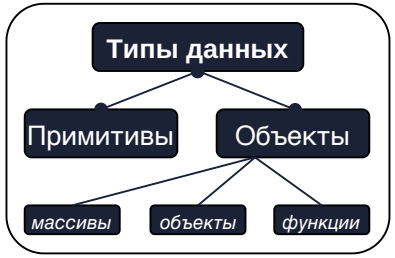
Примитивы

number число *(целое, с плавающей точкой)*	23 или 23.45 Специальные числовые значения: Infinity, -Infinity, NaN вычислительная ошибка (Not a Number)
string строка	Три типа кавычек: 1) 'одинарные' 2) "двойные" 3) `обратные *(допускают встраивание ${выражений} в строку и перенос строк)*`
boolean	*логический тип данных:* **true / false**
undefined	Переменная объявлена, но значение не присвоено
null	Значение не известно, пустое значение
bigint	9007199254740992**n** *(больше 2^{53}, меньше -2^{53})*
symbol	Уникальный идентификатор

Объекты *(структуры)*

array массив - *разновидность объекта, набор элементов с одинаковой структурой.* **Индекс элемента - индекс:** 0,1,2,3...	Два способа создания массива: 1) const **points = ['Bergen', 'Flam', 'Oslo']** 2) const **points = new Array ('Bergen', 'Flam', 'Oslo')** *Обратиться к элементу массива по индексу:* **points[0]** *Заменить элемент:* **points[2]='Stavanger'** *Узнать длину массива:* ***points.length*** *Преобразовать элемент массива в верхний регистр:* ***points[2].toUpperCase()***
object объект *(группировка значений с разной структурой, элементы объекта называются свойства, каждое свойство имеет **ключ и значение**)*	const **person = {** key: value, name: 'Lena', langudges = ['ru','en','hi','ar'], greet: function() {console.log ('Здравствуй!') } **}** *Обращение к свойствам объекта:* **person.name, person['name']** *Обращение к элементу массива в составе объекта:* **person.languages[3]** *Длина массива в составе объекта:* **person.languages.length** *Запуск функции (метода объекта):* **person.greet()**
function функция	Способы объявления функции: 1) Function declaration - работает в любом месте кода, независимо от того, где функция была объявлена **function myFunction () {}** 2)Function expression - работает с того места кода, где функция была объявлена *(такая функция считается анонимной, т.к. имени нет)* **const myFunction = function () {}**

3) Стрілочні функції (e) => {a+b}
немає свого контексту (this)



Методы встроенного объекта String

'текст'**.toLowerCase()** *в нижний регистр*
'текст'**.toUpperCase()** *в верхний регистр*
'текст'**.length** *число символов*
'текст'**.includes('ст')** *включает 'ст'? true/false*
'текст'**.indexOf('!')** *индекс элемента, нет: -1*

Методы встроенного объекта Array

points.push('элемент') добавить элемент в конец массива points.pop() удалить последний элемент

Метод map - копия массива с преобразованием:

const points = ['Bergen', 'Flam', 'Oslo']
pointsNew = points.**map(function** (*point*) {return *point* +'_New'})
console.log (pointsNew[0])//'Bergen New',
где point - переменная созданная внутри функции для обозначения элементов массива

Метод forEach - перебор массива:

Функция будет вызвана для каждого элемента массива:
const points = ['Bergen', 'Flam', 'Oslo']
points.**forEach**(**function** (item, index, array) {console.log*(item+', индекс:'+index)*}), *где item, index, array - новые переменные, созданные внутри функции*

Методы встроенного объекта Math

объект Math предназначен для математических вычислений:
Math.random() возвращает случайной число из диапазона [0;1) // 0,4007982330516835
Math.floor(1.21)//1 **Math.ceil(1.21)**//2
Math.max() *возвращает наибольшее значение*

Условный оператор if

if (условие) {команды}
else if (условие) {команды}
else if (условие) {команды}
...
else {команды}

Необязательные блоки

Тернарный оператор

(...) ? ... : ...
(условие) ? инструкция, если условие истинно : инструкция, если условие ложно

Циклы For *(перебор массива)*

for (let i = 0; i > cars.length; i++)
{*const car = cars [i] console.log (car)***}**
for (начало; условие; шаг) {тело цикла}

For of *(выводит элементы массива)*

for (let car **of** casrs**){**console.log (car)**}**
где car - новая переменная, созданная внутри цикла, cars - массив

do ... while

do {*тело цикла***} while (***условие***)**
Цикл выполнится, как минимум, 1 раз

Методы встроенного объекта Date

const now = **new Date ()** *текущая дата и время: //Thu May 28 2020 17:17:38 (Москва, стандартное время)*
new Date ('2020-05-28'**)** *задать дату*
now.toLocaleString('ru') *преобразовать в ru // 28.05.2020, 10:17:38*
Подробнее: https://learn.javascript.ru/date

Операторы

здесь приведены основные операторы, полный список ссылке: mdn

Доступ к свойствам	... **.** ... *точечная нотация* ... **[** ... **]** *скобочная нотация*		
Инкремент *(только для переменных)*	**i++** *(постфиксный)* +1, возвращает старое значение **++i** *(префиксный)* +1, возвращает новое значение		
Декремент *(только для переменных)*	**i--** *(постфиксный)* -1, возвращает старое значение **--i** *(префиксный)* -1, возвращает новое значение		
Унарный +, Унарный -	**+a** преобразует значение в число **-a** меняет знак числа на противоположное		
Логические	**II** или	**&&** и	**!** не
Математические	**+ - * / возведение в степень **** ***Внимание:*** *если при сложении хотя бы один операнд является строкой, то второй будет преобразован к строке*		
Сравнение	**==** равно по значению, **===** по значению и типу, **!=** не равно **>** больше, **>=** , **<** меньше, **<=**		
Присваивание	**=** присваивание, **+=** присваивание со сложением (a += b, смысл a = a + b) *есть также операторы присваивания с вычитанием, умножением и т.д.*		
typeof	Определяет тип данных: number, bigint, string, boolean, undefined, null, symbol, object, function		
return	завершает функцию и возвращает значение		

Конкатенация, встраивание и преобразование

Конкатенация - склеивание объектов линейной структуры.
"строка1" + 'СТРОКА2' = строка1СТРОКА2

Примечание: если оба операнда являются числами, то будет выполнено математическое сложение, если хотя бы один операнд является строкой, то второй будет преобразован к строке.
Пример: 3 + 3 + 'строка' = 6строка'

Встраивание выражений в текст:
`обратные кавычки допускают встраивание **${выражений}** в строку и **перенос** строк`

Допустимы переменные ${name}, функции ${age()}, тернарные выражения ${age<20 ? 'A' : 'B'}

Преобразование к числу:
+ a *оператор* унарный плюс преобразует значение в число

parseInt() *глобальная функция* преобразует значение в целое число, *округление вниз*
parseInt('56.78') // 56

parseFloat() *глобальная функция* преобразует значение в число с плавающей точкой
parseFloat('56.78') // 56.78

Примечание: такой же функционал имеют методы стандартного встроенного объекта Number: Number.parseInt() и Number.parseFloat()