**ES6 Object And Arrays Features**

**Синтаксический сахар** (syntactic sugar) - это синтаксические возможности, применение которых не влияет на поведение программы, но делает использование языка более удобным для человека.

Всё, что тут написано, это синтаксический сахар.

**Spread оператор**

Первым делом вспомним **Spread оператор**, который копирует все поля одного объекта в другой.

Он может работать как с объектами, так и с массивами. В первом случае копирует поля, во втором копирует элементы.

При копировании элементов, можно добавлять новые.

const array = [1,2,3]

//скопируем все элементы и добавим 4, 5

const array1 = [ ...array, 4, 5 ]

const obj = { marque: 'Ford', name: 'Mustang' }

//скопируем все поля и добавим поле production

const obj1 = { ...obj, production: 'March 1964 – present' }

**Rest оператор**

Несмотря на то, что синтаксис Rest оператора напоминает Spread, назначение первого отличается.

**Rest оператор** используется для доставания значений из массива.

В каждую указанную переменную будет записываться значение в зависимости от индекса, а оставшиеся значения запишутся в новый массив.

const array = [1,2,3,4,5]

const [firstItem, secondItem, ...otherArray] = array

console.log(firstItem) //1

console.log(secondItem) //2

console.log(otherArray) //[3,4,5]

Что интересно, Rest оператор работает и с аргументами функции.

Как мы знаем, в **Arrow** функции нет объекта аргументов **arguments**, но при этом мы можем использовать Rest оператор и достать все аргументы в отдельный массив, что будет равносильно объекту arguments.

function foo(item, ...args) {

console.log(item)

console.log(args)

}

foo(1,2,3,4)

*item и args - просто имена переменных, они могут быть любыми!*

**Ленивая инициализация аргументов**

В ES6 появилась возможность задавать базовые значения для аргументов. Это называется ленивой инициализацией.

Если наша функция ждёт какое-то значение в аргументе, но при её вызове в аргумент ничего не передают, то аргументу автоматически присваивается базовое значение.

function foo2(item = 'empty') {

console.log(item)

}

foo2()

foo2('abra')

**Shorthand (метод сокращения)**

Для сокращения записи объектов, в ES6 появилась возможность легко добавлять поля со значением, ключ которых совпадает с какой-то переменной.

const title = "mr"

const obj = {

title, //появилось поле title: "mr"

name: 'Igor'

}

console.log(obj)

Как вы видите, мы просто записали в объект переменную, имя которой стало ключем, а значение переменной стало значением поля.

При записи внутри объекта функции, Shorthand будет работать аналогичным образом

const obj = {

func: function() { },

func2() { }

}

console.log(obj)