Лекция №11

Массивы

Array

Array.prototype.slice()

Описание

Метод `slice()` не изменяет исходный массив, а возвращает новую «одноуровневую» копию, содержащую копии элементов, вырезанных из исходного массива. Элементы исходного массива копируются в новый массив по следующим правилам:

- Ссылки на объекты (но не фактические объекты): метод `slice()` копирует ссылки на объекты в новый массив. И оригинал, и новый массив ссылаются на один и тот же объект. То есть, если объект по ссылке будет изменён, изменения будут видны и в новом, и в исходном массивах.
- Строки и числа (но не объекты `String` и `Number`): метод `slice()` копирует значения строк и чисел в новый массив. Изменения строки или числа в одном массиве никак не затрагивает другой.

Array.prototype.splice()

Сводка

Метод `splice()` изменяет содержимое массива, удаляя существующие элементы и/или добавляя новые.

Array.prototype.reverse()

Интерактивный пример

Метод `reverse()` на месте обращает порядок следования элементов массива. Первый элемент массива становится последним, а последний — первым.

Array.prototype.concat()

Интерактивный пример

Метод `concat()` возвращает новый массив, состоящий из массива, на котором он был вызван, соединённого с другими массивами и/или значениями, переданными в качестве аргументов.

Array.prototype.join()

Сводка

Метод `join()` объединяет все элементы массива (или массивоподобного объекта) в строку.

Array.prototype.at()

Метод `at()` принимает значение в виде целого числа и возвращает элемент массива с данным индексом. В качестве аргумента метод принимает положительные и отрицательные числа. При отрицательном значении отсчёт происходит с конца массива.

Получение элементов массива с помощью квадратных скобок по-прежнему остаётся корректным способом. Например, `array[0]` вернёт первый элемент. Однако, при работе с элементами в конце массива больше нет необходимости прибегать к `array.length`. Например, для получения последнего элемента, вместо `array[array.length-1]` можно вызвать `array.at(-1)`. (Смотрите примеры ниже)

Array.prototype.forEach()

Описание

Метод `forEach()` выполняет функцию `callback` один раз для каждого элемента, находящегося в массиве в порядке возрастания. Она не будет вызвана для удалённых или пропущенных элементов массива. Однако, она будет вызвана для элементов, которые присутствуют в массиве и имеют значение `undefined`.

MAP

Описание

Метод `map` вызывает переданную функцию `callback` один раз для каждого элемента, в порядке их появления и конструирует новый массив из результатов её вызова. Функция `callback` вызывается только для индексов массива, имеющих присвоенные значения, включая `undefined`. Она не вызывается для пропущенных элементов массива (то есть для индексов, которые никогда не были заданы, которые были удалены или которым никогда не было присвоено значение.

Функция `callback` вызывается с тремя аргументами: значением элемента, индексом элемента и массивом, по которому осуществляется проход.

FILTER

Описание

Метод `filter()` вызывает переданную функцию `callback` один раз для каждого элемента, присутствующего в массиве, и создаёт новый массив со всеми значениями, для которых функция `callback` вернула значение, которое может быть приведено к `true`. Функция `callback` вызывается только для индексов массива с уже определёнными значениями; она не вызывается для индексов, которые были удалены или которым значения никогда не присваивались. Элементы массива, не прошедшие проверку функцией `callback`, просто пропускаются и не включаются в новый массив.

REDUCE

Описание

Метод `reduce()` выполняет функцию `callback` один раз для каждого элемента, присутствующего в массиве, за исключением пустот, принимая четыре аргумента: начальное значение (или значение от предыдущего вызова `callback`), значение текущего элемента, текущий индекс и массив, по которому происходит итерация.

При первом вызове функции, параметры `accumulator` и `currentValue` могут принимать одно из двух значений. Если при вызове `reduce()` передан аргумент `initialValue`, то значение `accumulator` будет равным значению `initialValue`, а значение `currentValue` будет равным первому значению в массиве. Если аргумент `initialValue` не задан, то значение `accumulator` будет равным первому значению в массиве, а значение `currentValue` будет равным второму значению в массиве.

FIND

Описание

Метод `find` вызывает переданную функцию `callback` один раз для каждого элемента, присутствующего в массиве, до тех пор, пока она не вернёт `true`. Если такой элемент найден метод `find` немедленно вернёт значение этого элемента. В противном случае, метод `find` вернёт `undefined`. До Firefox 34 функция `callback` не вызывалась для «дырок» в массивах (bug 1058394).

Функция `callback` вызывается с тремя аргументами: значением элемента, индексом элемента и массивом, по которому осуществляется проход.

SOME

Описание

Метод `some()` вызывает переданную функцию `callback` один раз для каждого элемента, присутствующего в массиве до тех пор, пока не найдёт такой, для которого `callback` вернёт *истинное* значение (значение, становящееся равным `true` при приведении его к типу `Boolean`). Если такой элемент найден, метод `some()` немедленно вернёт `true`. В противном случае, если `callback` вернёт `false` для всех элементов массива, метод `some()` вернёт `false`. Функция `callback` вызывается только для индексов массива, имеющих присвоенные значения; она не вызывается для индексов, которые были удалены или которым значения никогда не присваивались.

EVERY

Описание

Метод `every()` вызывает переданную функцию `callback` один раз для каждого элемента, присутствующего в массиве до тех пор, пока не найдёт такой, для которого `callback` вернёт *ложное* значение (значение, становящееся равным `false` при приведении его к типу `Boolean`). Если такой элемент найден, метод `every()` немедленно вернёт `false`. В противном случае, если `callback` вернёт `true` для всех элементов массива, метод `every()` вернёт `true`. Функция `callback` вызывается только для индексов массива, имеющих присвоенные значения; она не вызывается для индексов, которые были удалены или которым значения никогда не присваивались.

Array.prototype.flat()

Метод `flat()` возвращает новый массив, в котором все элементы вложенных подмассивов были рекурсивно "подняты" на указанный уровень depth.

Array.prototype.sort()

Сводка

Метод `sort()` *на месте* сортирует элементы массива и возвращает отсортированный массив. Сортировка не обязательно устойчива (англ.). Порядок сортировки по умолчанию соответствует порядку кодовых точек Unicode.

Полезные ссылки

https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Global_Objects/Array#methods

Спасибо!