Cheatography

CoreJS Interview1 Cheat Sheet

by itsMe via cheatography.com/189476/cs/39515/

JS Basics

Falsy ⇒ '', 0, null, undefined, NaN, fals

var - hoisting, let, const - blocks

Temporal Dead Zone начинается в начале области видимости переменной и заканчивается ее объявлением

Primitives: `Undefined, Boolean, String, Symbol, BigInt, Null`

Non Primotives:
Object, Array, Function, RegExp, Date, Map, Set,
Error, Promise, enum

Strings

`text.length`

`text.slice(start, end)` => new string [start, end)

`text.substring( start, end)` new string [start, end)

`text.substr( start, length)` new string [start... `length` number of characters.

`text.replace (oldValue, newValue) text.replaceAll (oldValue, newValue)`

`text.toUpperCase() text.toLowerCase()`

`concat (string1, string2, ..., stringN)`

`text.trim() trimStart() trimEnd()`

`padStart( targetLength, padString)`: adds padding to the beginning

`padEnd (targetLength, padString)`:adds padding to the end

`charAt (index) charCodeAt(index)`

`split( separator, limit)`

Timers

Методы setInterval(func, delay, ...args) и setTimeout(func, delay, ...args) позволяют выполнять func регулярно или только один раз после задержки delay, заданной в мс.

ля отмены выполнения необходимо вызвать clearInterval/clearTimeout со значением, которое возвращают методы setInterval/setTimeout.

Timers (cont)

Планирование с нулевой задержкой setTimeout(func,0) или, что то же самое, setTimeout(func) используется для вызовов, которые должны быть исполнены как можно скорее, после завершения исполнения текущего кода.

Браузер ограничивает 4-мя мс минимальную задержку между пятью и более вложенными вызовами setTimeout, а также для setInterval, начиная с 5-го вызова.

```
let timerId = setTimeout(...); clearTimeout(timerId);
```

Objects

Creating Objects

Literal/ Contructor Function/ Class

`{x: " y"} (new) function(){} new Object() same as {} class Banana {} Object.create(proto[, propertiesObject])`

`delete book.userSReading`

Обход объекта

`for (let key in myObject) Object.entries() => [key, value]`

`Object.keys() ⇒ [key] Object.values() => [values]`

Методы

`someObj.hasOwnProperty('key1')`

`obj.__proto__ === person ⇒ Object.getPrototypeOf (object1) Object.setPrototypeOf (object1)`

The `valueOf()` method of `[Object]` instances converts the `this` value [to an object]

Object.create()

```
function MyClass() {
    SuperClass.call (this);
    OtherSuperClass.call( -this);
}
MyClass.prototype = Object.create(SuperClass.prototype);
```

позволяет создавать новый объект и устанавливать прототип этого объекта в объект, переданный в качестве параметра

Events

Event Phases - три фазы жизненного цикла события: захват (capturing), цель (target) и всплытие (bubbling).

Разница между этими фазами заключается в порядке вызова обработчиков событий. В фазе захвата обработчики вызываются сверху вниз, а в фазе всплытия - снизу вверх. В фазе цели вызывается только один обработчик, связанный с элементом цели.

Mouse Events: - click - dblclick - mousedown - mouseup - mousemove - mouseover - mouseout - contextmenu

Keyboard Events: - keydown - keyup - keypress

Form: formdata reset submit;

Input:change, input, cut, copy, paste

Array NOT Mutating

`concat(array1, array2, ..., arrayN`

`array.filter (callback(el, ind, Array`

`array.map (callback (elt, ind, array`

`array.slice( start, end): new Ar`

`array.reduce (callback (acc, cany`

`reduceRight( callback(acc, cur`

Array Mutations

`push(element1, element2, ..., el`

`pop()`

`shift()`

`unshift(element1, element2, ...`

`splice (start, deleteCount, item1 N)`

`sort(compareFunction)`

`reverse()`

`fill(value, start, end)`

Iterate Array

```
for (let i = 0; i <
scores.length; i++) {}
for (i in scores) {}
for (score of scores) {}
scores.fo rEa ch( (score) =>
{});
every( fun ction (element,
index, array) { / … / },
thisArg)
array.r educe( functi on( total,
el, i, arr),i nit ial Value )
every( fun ction (element,
index, array) { / … / },
thisArg)
```

Copy Array

```
const arrNew = [...arrOld];
const newArray = [...ol dAr -
ray];
{ const json = JSON.s tri ngi -
fy( old Array);
const newArray = JSON.p ars -
e(j son); }
const arrNew = arrOld.sl ice();
const arrNew = Array.f ro m(a -
rrOld);
const arrNew = [].con cat (ar -
rOld);
```

By **itsMe**
cheatography.com/itsme/

Published 9th July, 2023.
Last updated 9th July, 2023.