10 самых частоиспользуемых функций <http://javascript.ru/unsorted/top-10-functions>

# Получение случайного числа в заданном диапазоне (random, разброс критов)

var min = 1; max = 3; //вводимые минимальные и максимальные числа

function random(min,max){

var result = Math.random () \* (max - (min - 1)); //включая мин и макс значения

return parseInt(result) + min; //округление числа до целого

}

Случайное число в диапазоне от 0 до 10

**math.floor(math.random() \* 11**) //возвращает рандом от 0 до 10

Получения текущего дня недели

var d = new Date(); //тукущая дата

var days = [“Sun”, Mon”, “Tue”, “Wed”, “Thu”, “Fri”, “Sat”]; //массив с 0 от вск

document.getElementById(“demo”).innerHTML = days[d.getDay] //возвращает строку

Проверка переменной на массив

Function isArray(myArray){ //подаем массив на проверку

Return myArray.constructor.toString().indexOf(“Array”) > -1;

} //сравнивает

# Быстро скопировать массив

var arr = [ "HTML", "JavaScript", "CSS" ]

var newArr = arr.slice()

# **Функция сортировки чисел от большего к меньшему**

function compareNumeric(a, b) {

return a - b;

}

var arr = [ 1, 2, 15 ];

arr.sort(compareNumeric); //указываем функцию без скобок () !!

alert(arr); // 1, 2, 15

/\*\*\*\*\*тоже самое, только применимо к массиву данных с возрастом.\*\*\*\*\*/

var vasya = {name: "Вася", age: 23};

var masha = {name: "Маша", age: 18}; //обьекты с со свойствами

var vovochka = {name: "Вовочка", age: 6};

var people = [vasya, masha, vovochka]; //массив с обьектами со свойствами

function sortedByAge(personA, personB) {

return personA.age - personB.age; //неизвестные данные с известным свойством

}

people.sort(sortedByAge); // отсортирует по возрасту, от большего к меньшему

# Проверка на число.

function isNumeric(n) {

return !isNaN(parseFloat(n)) && isFinite(n)

}

# **Вложенные списки (вывод односвязного списка)**

var list = {

value: 1, next: {

value: 2, next: {

value: 3, next: {

value: 4, next: null

}

}

}

};

function printList(list) {

var tmp = list; /\*\*\*временная переменная используется для того, что бы мы в будущем не сможем расширить функцию и сделать со списком что-то ещё\*\*\*/

while(tmp) {

alert( tmp.value );

tmp = tmp.next;}

}

# Умножение и перезапись свойств обьекта в два раза:

var menu = {

width: 200,

height: 300,

title: "My menu"

};

function isNumeric(n) { //проверка на число

return !isNaN(parseFloat(n)) && isFinite(n)

}

function multiplyNumeric(obj){

for(var key in obj){ //включаем счетчик свойств из переданного обьекта

if (isNumeric(obj[key])){ //проверяем, численное ли значение свойства

obj[key] \*= 2; //перезаписывает свойство

}

}

}

multiplyNumeric(menu);

alert("menu width="+menu.width+" height="+menu.height+" title="+menu.title);

# Поиск и замена не уникальных (повторяющихся значений):

**function unique(str) {**

var arr = str.split(“ “)

/\* смысл в том, что бы привести все значения в свойства объекта. Объект не-уникальные свойства заменяет. Split сделает из строки массив со свойствами\*/

**var obj = {};** //создаем пустой обьект, который будет заменять свойства

**for(var i=0; i<arr.length; i++) {** //счетчик будет перебирать

**var n = arr[i];** //переобразуем значение для наглядности

**obj[n] = true;** // запомнить строку в виде свойства объекта

**}**

**return Object.keys(obj);** // или собрать ключи перебором для IE<9

**}**

# **Узнать, какой день, число был в прошлом или будет в будущем**

function dateDetect(dayN) { //заходит день

var date = new Date(); //определяем дату текущего дня

date.setDate(dayN); //назначаем новую дату, автоисправление подстраивает

var dayArr = ["вс", "пн", "вт", "ср", "чт", "пт", "сб"]; //вс = [0]

return dayArr[(date.getDay())]; //возвращает в днях недели

}

console.log(-100) // минус 100 дней назад

Бенчмаркинг, тест на время, тестирование

function floorMath(x) { return Math.floor(x); } //первая функция,которая тестируется

function floorXor(x) { return x^0; } //вторая функция, которая тестируется

function bench(func) {

var date = new Date(); //задаем текущее время

for (var i=0.5; i<100000; i++) func(i); //прогоняем каждую функцию

return new Date() - date; //отнимается время

}

var timeMath = 0, timeXor = 0;

for(var i=0; i<100; i++) {

timeMath += bench(floorMath); //еще больше прогона

timeXor += bench(floorXor);

}

console.log('Время floorMath: ' + timeMath + 'мс');

console.log ('Время floorXor: ' +timeXor + 'мс');

**Обьект счетчика + функция**

**В данном примере создается обьект с функциями внутри, которые могут быть методами обьекта**

****function makeCounter() {****

****var currentCount = 0;****

****function counter() {**** // возвращаемся к функции

****return ++currentCount;****

****}****

****counter.set = function(value) {**** // ...и добавляем ей методы!

****currentCount = value;****

****};****

****counter.reset = function() {****

****currentCount = 0;****

****};****

****return counter;****

****}****

****var counter = makeCounter();****

****counter.set(5)****

****console.log(counter());**** //6. Обьект вызывается как функция

Счетчик внутри функции

function makeCounter() { //счетчик

return function f() { //возвращаемая функция

if (!f.currentCount){ //если счетчик не назначен

f.currentCount = 0 //назначить нулем

}

return f.currentCount++; //вернуть счетчик

};

}

var c1 = makeCounter(); //при вызове счетчик начинает отсчет

var c2 = makeCounter(); //не зависим от первого счетчик

console.log(c1())

console.log(c2())

Именованные аргументы (несколько аргументов в функции)

/\*Функция **по умолчанию** \*/

function f(option) {

var width = option.width || 200;

var height = option.height || 100; //задается значение ИЛИ оно 100

var complete = "Width: " + width + "\n" + "Height: " + height;

return complete;

}

var option = { //задаем опции

width: 140,

height: null

}

console.log(f(option));

showWarning(opts); // не нужно копировать остальные аргументы в вызов

**Сделать из arguments аргументов массив**

function printArgs() {

// вызов arr.slice() скопирует все элементы из this в новый массив

var args = [].slice.call(arguments);

alert( args.join(', ') ); // args - полноценный массив из аргументов

}

printArgs('Привет', 'мой', 'мир'); // Привет, мой, мир

**slice** для работы требует только нумерованные свойства и **length**. Всё это в **arguments** есть.

**Передача аргументов из внешнего лексического окружения, передать в функцию, достать аргумент, задержка, интервал, счетчик**

function delay(f, ms) {

return function() { //будем возвращать обертку

var savedThis = this; //записываем контекст обьекта

var savedArgs = arguments; //записываем аргументы

setTimeout(function() { //назначем задержку

f.apply(savedThis, savedArgs);//передаем обьект и аргументы в функцию f

}, ms);

};

}

function f(x) {

alert( x );

}

var f1000 = delay(f, 1000);

var f1500 = delay(f, 1500);

f1000("тест"); // выведет "тест" через 1000 миллисекунд

f1500("тест2"); // выведет "тест2" через 1500 миллисекунд

Добавление нового элемента в цикле в HTML (копирование элемента)

//создаем элемент (пока не в ДОМ)

var closeButton = document.createElement("button");

closeButton.className = "remove-button"; //кнопка для закрытия элементов

closeButton.innerHTML = "[x]";

closeButton.style.position = "absolute";

closeButton.style.right = "0";

closeButton.style.top = "0";

//берем контейнеры, куда пихаем кнопку

var containers = document.getElementsByClassName("pane");

//копируем методы из массива

containers = Array.prototype.slice.call(containers);

containers.forEach(function(container){

container.style.position = "relative";

//делаем глубокую копию элемента, иначе вставиться только один элемент

var closeButtonCopy = closeButton.cloneNode(true);

container.appendChild(closeButtonCopy);

});

**Элемент увеличивает высоту от величины экрана для правильного позиционирования на фоновом изображении**

window.onresize = resize;

function resize(){

var width = document.body.clientWidth;

var topBlock = document.getElementsByClassName("logo-phone")[0];

if(width > 1101) {

topBlock.style.height = ((width / 2) - 300) + "px";

}else {

topBlock.style.height = "260px";

}

}

window.onload = resize();

Прижимаем футер к низу контента, в зависимости от количества контента

var content = document.querySelector("main");

var header = document.querySelector("header");

var footer = document.querySelector("footer");

function contentMinHeight(){

var windowHeight = document.documentElement.clientHeight;

var headerHeight = header.offsetHeight + parseInt(getComputedStyle(header).margin);

var contentHeight = content.offsetHeight + parseInt(getComputedStyle(content).margin);

var footerHeight = footer.offsetHeight + parseInt(getComputedStyle(footer).margin);

if((contentHeight + footerHeight + headerHeight) < windowHeight) {

content.style.minHeight =

(windowHeight - headerHeight - footerHeight - parseInt(getComputedStyle(content).margin)) + "px";

}

}

window.onload = contentMinHeight();

window.onresize = contentMinHeight;

Аккордион на js

var menu = document.querySelector("menu"); //выбираем все меню

function hide(parent) {

var headers = parent.querySelectorAll("li > ul"); //выбираем детей

for (var i = 0; i < headers.length; i++) {

headers[i].style.display = "none"

}

}

hide(menu); //скрываем при загрузке страницы

menu.addEventListener("click", showHide); //вешаем обработчик на все меню

function showHide(event){

var header = event.target;

var parent = header.parentElement; //выбираем родителя события

hide(parent); скрываем детей родителя

header.querySelector("ul").style.display = ""; //открываем текущий элемент

}

Мой не-адаптивный слайдер

/\*SLIDER\*/

function slider(timer) {

var sliderBlock = document.querySelector(".slider-wrapper");

function \_parentWidth(){

var wrapperWidth = sliderBlock.children[0].offsetWidth \* 2; //что бы было пространство для слайда

sliderBlock.style.width = wrapperWidth + "px";

};

function \_slide(){

setTimeout(function(){

var \_firstImg = sliderBlock.children[0];

\_firstImg.style.marginLeft = "-" + 1100 + "px"; /\*ширина картинки задается тут\*/

counter(); // для смены нумерации слайдов, при ненадобности убрать

\_remove(\_firstImg);

}, timer)

}

function \_remove(\_firstImg){

\_firstImg.addEventListener("transitionend", function(){

/\*по-другому не получалось удалить и вставить элемент\*/

var \_removedImg = \_firstImg.cloneNode(true);

sliderBlock.removeChild(\_firstImg);

\_removedImg.style.marginLeft = '';

sliderBlock.appendChild(\_removedImg);

console.log(\_firstImg);

\_slide(); /\*зацикливается тут, как сделать адекетный (точный) setInterval не понял\*/

})

}

function \_countSlides() {

/\*функция подсчитывает количество изображений и строит соответственное число кружочков-переключателей\*/

var count, countParent;

count = sliderBlock.children;

countParent = document.querySelector(".counter-slide");

var li = document.createElement("li");

li.classList.add("counter\_\_item");

for (var i = 0; i < count.length; i++) {

countParent.appendChild(li.cloneNode(true));

}

countParent.children[0].classList.add("counter\_\_item--active")

}

function \_changeCountSlides(){

/\*Типичная функция счетчика по вызову\*/

var startCounter = 0;

var counters = document.querySelector(".counter-slide").children;

return function(){

startCounter++;

console.log(startCounter);

startCounter >= (counters.length) ? startCounter = 0 : true;

var currentCounter = document.querySelector(".counter\_\_item--active");

currentCounter.classList.remove("counter\_\_item--active");

return counters[startCounter].classList.add("counter\_\_item--active");

}

}

var counter = \_changeCountSlides();

window.addEventListener("load", \_parentWidth);

\_countSlides();

\_slide();

}

window.addEventListener("load", slider(4000));

СЛАЙДЕР С АБСОЛЮТНОПОЗИЦИОНИРОВАННЫМИ БЛОКАМИ

сделан через transform - в 3 раза лучше для процессора

**function Slider(sliderId) {  
 var slider, slides, currentSlide, nextSlide, prevSlide, slideWidth, slideVisible;  
  
 /\*сlasses can be defined by the arguments\*/  
 slider = document.querySelector(sliderId);  
 slider.style.position = "relative"; //definitely positioned  
 slides = slider.children;  
 slideWidth = "1200px";  
 slideVisible = "js\_\_slider-image--visible";  
 currentSlide = "js\_\_slider-image--current";  
 nextSlide = "js\_\_slider-image--next";  
 prevSlide = "js\_\_slider-image--prev";  
  
 //adding classes  
 slides[0].className += currentSlide; //not classList!! Not working in ie8  
 for (var i = 0; i < slides.length; i++) {  
 slides[i].className += " js\_\_slider-image";  
 }  
  
  
 //next button for loop  
 function next() {  
  
 if (window.addEventListener) {  
  
 var index = 0;  
 while (!slides[index].classList.contains(currentSlide)) {  
 index++;  
 }  
  
 /\*animation through transformation to not overload the processor\*/  
 if (index < slides.length - 1) {  
 slides[index + 1].classList.add(slideVisible);  
 slides[index + 1].style.transform = "translate(" + slideWidth + ", 0)";  
  
 setTimeout(function () {  
 slides[index].style.transform = "translate(-" + slideWidth + ", 0)";  
 slides[index + 1].style.transform = "translate(0, 0)";  
 }, 100);  
 } else {  
 slides[0].classList.add(slideVisible);  
 slides[0].style.transform = "translate(" + slideWidth + ", 0)";  
  
 setTimeout(function () {  
 slides[index].style.transform = "translate(-" + slideWidth + ", 0)";  
 slides[0].style.transform = "translate(0, 0)";  
 }, 100);  
 }  
  
 } else { //code for IE9-  
  
 }  
 }  
  
  
 //previous button for loop  
 function previous() {  
  
 if (window.addEventListener) {  
  
 var index = 0;  
 while (!slides[index].classList.contains(currentSlide)) {  
 index++;  
 }  
  
 if (index === 0) {  
 slides[slides.length - 1].classList.add(slideVisible);  
 slides[slides.length - 1].style.transform = "translate(-" + slideWidth + ", 0)";  
  
 setTimeout(function () {  
 slides[index].style.transform = "translate(" + slideWidth + ", 0)";  
 slides[slides.length - 1].style.transform = "translate(0, 0)";  
 }, 100);  
 } else {  
 slides[index - 1].classList.add(slideVisible);  
 slides[index - 1].style.transform = "translate(-" + slideWidth + ", 0)";  
  
 setTimeout(function () {  
 slides[index].style.transform = "translate(" + slideWidth + ", 0)";  
 slides[index - 1].style.transform = "translate(0, 0)";  
 }, 100);  
 }  
  
 } else { //code for IE9-  
  
 }  
 }  
  
 function clearSlides() {  
 var prevElem = slider.getElementsByClassName(currentSlide)[0];  
 var current = slider.getElementsByClassName(slideVisible)[0];  
  
 if(prevElem && current) {  
 setTimeout(function () {  
 current.classList.add(currentSlide);  
 current.style.transform = "";  
  
 setTimeout(function () {  
 prevElem.classList.remove(currentSlide);  
 current.classList.remove(slideVisible);  
 prevElem.style.transform = "";  
 }, 0);  
  
 }, 0);  
 }  
 }  
  
 slider.addEventListener("transitionend", clearSlides);  
  
 //loop for slider  
 var timerId = setTimeout(function tick() {  
 next();  
 timerId = setTimeout(tick, 6000);  
 }, 6000);  
}  
  
var mainSlider = new Slider("#main-slider");**

СТЕК ФУНКЦИИ ДЛЯ ПООЧЕРЕДНОГО ВЫЗОВА (jQuery)

function d1() {

$('#d1').slideUp(1000).slideDown(1000);

cb();

}

function d2() {

setTimeout(function(){

$('#d2').hide(1000).show(1000);

cb();

},2000);

}

function d3() {

$('#d3').fadeOut(1000).fadeIn(1000);

cb();

}

var fns = [ d1, d2, d3 ];

function cb()

{

var fn = fns.shift();

if( typeof fn == 'function' ) fn.call();

}

cb();

СТЕК ФУНКЦИИ ДЛЯ ПООЧЕРЕДНОГО ВЫЗОВА (js)

var show\_message = function () {

var stack = [];

var inProgress = false;

function animate() {

var currentMessage = stack.shift();

/\*

тут запускаешь свои анимации и

в коллбэке по завершению, запускаешь

следующую:

if (stack.length) animate();

else inProgress = false;

\*/

}

return function (message) {

stack.push(message);

if (!inProgress) {

inProgress = true;

animate();

}

};

} ();