

# Типы данных

Чистяков Денис

# Строки

*Строки полезны для хранения данных, которые можно представить в текстовой форме.*

Mozilla Developer Network

# Создание строки

```
// Пустая строка  
var emptyString = '';
```

```
// Длина строки  
emptyString.length; // 0
```

# Создание строки

```
// Можно использовать одинарные кавычки  
var russianString = 'строка текста';  
  
russianString.length; // 13
```

```
// Можно использовать двойные кавычки  
var russianString = "строка текста";  
  
russianString.length; // 13
```

# Создание строки

```
var escapeCodesString = 'a\'b' // a'b  
escapeCodesString.length; // 3
```

```
var escapeCodesString = 'a\\b' // a\b  
escapeCodesString.length; // 3
```

```
var escapeCodesString = 'a\n\tb' // a      b  
                             //      b  
escapeCodesString.length; // 4
```

# Создание строки

```
// Поддерживаются все символы из Unicode  
var utf8String = '中文 español English русский 日本 ਪੰਜਾਬ 한';  
utf8String.length; // 36
```

# Строки являются незаписываемыми

```
var russianString = 'кот';
```

```
// Возможно обращение к символу по индексу
```

```
russianString[1]; // 'о'
```

```
// Редактирование невозможно
```

```
russianString[1] = 'и';
```

```
russianString; // 'кот'
```

# Срезы из строк

```
// Обрезаем строку до 140 символов под длину твита
var longString = 'Очевидно проверяется, что математический анализ \
существенно масштабирует интеграл по поверхности, что неудивительно. \
Первая производная, очевидно, позиционирует ортогональный определитель.'

var shortString = longString;

if (longString.length > 140) {
    shortString = longString.slice(0, 139) + '...';
}



shortString; // 'Очевидно проверяется, что математический анализ
// существенно масштабирует интеграл по поверхности, что неудивительно.
// Первая производная, оч...'
shortString.length; // 140
```



# Поиск в строке

```
var tweet = 'PWA. Что это такое? Третий доклад на WSD в Питере Сергея ' +  
    'Густуна #wstdays';  
  
// Находим индекс первого вхождения подстроки в строке  
tweet.indexOf('#wstdays'); // 65  
  
// Искомая подстрока отсутствует  
tweet.indexOf('#fronttalks'); // -1
```

# Преобразование строки к числу

 USD ▾	100	×
 RUB ▾	5 914,76	×

по курсу ЦБ РФ на 26.08.2017

100\$ в рублях

# Преобразование строки к числу

```
var selector = '.converter-form .input .input__control';  
var usdInputAsText = document.querySelector(selector).value;
```

```
var usdInputAsInt = Number(usdInputAsText); // 100
```

```
var usdInputAsInt = parseInt(usdInputAsText, 10); // 100
```

```
var usdInputAsBin = parseInt(usdInputAsText, 2); // 4
```

```
var incorrect = parseInt('foo'); // NaN
```

# Массивы

*Массивы являются спископодобными объектами, чьи прототипы содержат методы для операций обхода и изменения массива. Ни размер JavaScript-массива, ни типы его элементов не являются фиксированными.*

Mozilla Developer Network

# Создание массива

```
// Пустой массив
var emptyArray = [];

emptyArray.length; // 0

// Массив чисел
var arrayOfNumbers = [1, 2, 3, 4];

arrayOfNumbers.length; // 4

// Массив строк
var arrayOfStrings = ['a', 'b', 'c'];

arrayOfStrings.length; // 3
```

# Итерирование по массиву

```
var tweets = [  
  'Я и IoT, пятый доклад на WSD в Питере Вадима Макеев #wstdays',  
  'Вёрстка писем. Развенчиваем мифы. Четвёртый доклад на WSD в Питере Артура Коха #wstda',  
  'PWA. Что это такое? Третий доклад на WSD в Питере Сергея Густуна #wstdays',  
  'Pokémon GO на веб-технологиях, второй доклад на WSD в Питере Егора Коновалова #wstday',  
  'Ого сколько фронтендеров. #wstdays',  
  '<head> – всему голова, первый доклад на WSD в Питере Романа Ганина #wstdays',  
  
  'Доброе утро! WSD в Питере начинается через 30 минут: программа, трансляция и хештег #',  
  'Наглядная таблица доступности возможностей веб-платформы Пола Айриша: Can I use + Sta',  
  'Node.js, TC-39 и модули, Джеймс Снел о проблемах Node.js с асинхронными модулями ES и',  
  'Всегда используйте <label>, перевод статьи Адама Сильвера в блоге Академии HTML',  
  'JSX: антипаттерн или нет? Заметка Бориса Сердюка на Хабре',  
  'Как прятать инлайновые SVG-иконки от читалок, Роджер Йохансен объясняет, зачем это ну',  
];  
  
tweets.length; // 12
```

# Итерирование по массиву

```
for (var i = 0; i < tweets.length; i++) {  
    var tweet = tweets[i];  
  
    // Что-то делаем с конкретным твитом  
}
```

# Добавление / удаление из массива

```
var emptyArray = [];  
  
// Добавляем элементы  
emptyArray.push('a');  
emptyArray.push('b');  
emptyArray; // ['a', 'b']  
  
emptyArray.length; // 2  
  
// Удаляем последний элемент  
emptyArray.pop(); // 'b'  
emptyArray; // ['a']  
  
emptyArray.length; // 1
```



# Объединение массивов

```
var concatedArray = arrayOfNumbers.concat(arrayOfStrings);
```

```
concatedArray; // [1, 2, 3, 4, 'a', 'b', 'c']
```

```
arrayOfNumbers; // [1, 2, 3, 4]
```

```
arrayOfStrings; // ['a', 'b', 'c']
```

# Фильтрация массива

```
// Найдем все строки, содержащие хештег #wstdays в массиве
var result = [];

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

    var tweet = tweets[i];

    if (tweet.indexOf('#wstdays') !== -1) {
        result.push(tweet);
    }
}
```

# Срезы массивов

```
// Выберем первые 5 твитов из массива  
var tweetsByPage = tweets.slice(0, 5);  
tweetsByPage.length; // 5
```

```
// Копируем оригинальную ленту в новый массив  
var tweetsWithAdv = tweets.slice();  
tweetsWithAdv.length; // 12
```

# Операции с массивом твитов

```
// Добавим немного рекламы в ленту, изменяя копию  
tweetsWithAdv.splice(4, 0, 'Покупайте наших слонов!');  
tweetsWithAdv.length; // 13
```

```
// Удалим несколько твитов из ленты  
tweetsWithAdv.splice(7, 2);  
tweetsWithAdv.length; // 11
```

```
tweets.length; // 12
```

# Объекты

*Список, состоящий из пар с именем свойства и связанного с ним значения, которое может быть произвольного типа.*

Mozilla Developer Network

# Создание объекта

```
// Пустой объект  
var emptyObject = {};
```

```
// Объект с predetermined набором свойств  
var tweet = {  
  createdAt: 'Sat Oct 01 12:01:08 +0000 2016',  
  id: 782188596690350100,  
  text: 'Я и IoT, пятый доклад на WSD в Питере Вадима Макеева #wstdays',  
  user: {  
    id: 42081171,  
    name: 'Веб-стандарты',  
    screenName: 'webstandards_ru',  
    followersCount: 6443  
  },  
  hashtags: ['wstdays']  
};
```

# Обращение к свойствам объекта

```
// Получение значения свойства через точечную нотацию  
emptyObject.propertyName = 'foo'; // { propertyName: 'foo' }  
emptyObject.propertyName; // 'foo'
```

```
// Удаление свойства  
  
delete emptyObject.propertyName; // {}
```

```
tweet.id; // 782188596690350100  
tweet.user.screenName; // 'webstandards_ru'
```

```
// Получение значения свойства через квадратные скобки  
tweet['i' + 'd']; // 782188596690350100
```

# Итерирование по ключам объекта

```
var keys = Object.keys(tweet);  
keys; // ['createdAt', 'id', 'text', 'user', 'hashtags']
```

```
for (var i = 0; i < keys.length; i++) {  
  var key = keys[i];  
  var value = tweet[key];  
  
  // Что-то делаем с ключом и со значением  
}
```



## Проверка наличия свойства у объекта

```
tweet.hasOwnProperty('text'); // true
```

```
tweet.hasOwnProperty('nonExistantProperty'); // false
```

# Функции

*Именованный блок кода, который позволяет переиспользовать существующий код.*

*Может иметь входные параметры и возвращать значение.*

*Является объектом высшего порядка.*

# Декларация функции

```
function getFollowersCount() {  
    return 6443;  
}
```

# Декларация функции

```
function noop() {  
}
```

# Декларация функции

```
function noop() {  
    return undefined;  
}
```

# Декларация функции

```
function getAuthor(tweet) {  
    return tweet.user.screen_name;  
}
```

# Передача по значению

```
tweet.user.followersCount; // 6443

function incrementFollowersCount(count) {
  count++; // 6444
}

incrementFollowersCount(tweet.user.followersCount);

tweet.user.followersCount; // 6443
```

# Передача по ссылке

```
tweet.user; // { id: 42081171, name: 'Веб-стандарты',  
// screenName: 'webstandards_ru', followersCount: 6443 }  
  
function incrementFollowersCount(user) {  
    user.followersCount++;  
}  
  
incrementFollowersCount(tweet.user);  
tweet.user; // { id: 42081171, name: 'Веб-стандарты',  
// screenName: 'webstandards_ru', followersCount: 6444 }
```



*Функции — объекты высшего порядка.*

Они могут быть переданы в другие функции в качестве аргумента, а так же могут иметь личные свойства, как и другие объекты.

# Передача функций в качестве аргументов

```
var tweets = [  
  { hashtags: ['wstdays'], likes: 16, text: 'Я и IoT, пятый...' },  
  { hashtags: ['wstdays', 'mails'], likes: 33, text: 'Вёрстка писем...' },  
  { hashtags: ['wstdays'], likes: 7, text: 'PWA. Что это...' },  
  { hashtags: ['wstdays', 'pokemongo'], likes: 12, text: 'Pokémon GO на...' },  
  { hashtags: ['wstdays'], likes: 15, text: 'Ого сколько фронт...' },  
  
  { hashtags: ['wstdays', 'html'], likes: 22, text: '<head> — всему...' },  
  { hashtags: ['wstdays'], likes: 8, text: 'Доброе утро! WSD...' },  
  { likes: 9, text: 'Наглядная таблица доступности...' },  
  { hashtags: ['nodejs'], likes: 7, text: 'Node.js, TC-39 и модули,...' },  
  { hashtags: ['html'], likes: 28, text: 'Всегда используйте <label>...' },  
  { likes: 18, text: 'JSX: антипаттерн или нет?...' },  
  { hashtags: ['svg'], likes: 19, text: 'Как прятать инлайновые...' }  
];
```

# Операции с массивом. forEach

```
var result = [];  
  
// Теперь в result лежат отфильтрованные твиты  
tweets.forEach(filterWithWstdaysHashtag);  
  
// Выбираем только твиты с хештегом #wstdays  
function filterWithWstdaysHashtag(tweet, index) {  
    var hashtags = tweet.hashtags;  
    if (Array.isArray(hashtags)  
        && hashtags.indexOf('wstdays') !== -1) {  
        result.push(tweet);  
    }  
}
```

# Операции с массивом. forEach под капотом

```
function forEach(callback) {  
    for (var i = 0; i < this.length; i++) {  
        callback(this[i], i);  
    }  
}
```

```
function filterWithWstdaysHashtag(tweet, index) {  
    var hashtags = tweet.hashtags;  
    if (Array.isArray(hashtags)  
        && hashtags.indexOf('wstdays') !== -1) {  
        result.push(tweet);  
    }  
}
```

```
forEach(filterWithWstdaysHashtag);
```

# Операции с массивом. filter

```
// Теперь в filteredTweets лежат отфильтрованные твиты
var filteredTweets = tweets.filter(isWstdaysHashtagExists);

// Выбираем только твиты с хештегом #wstdays
function isWstdaysHashtagExists(tweet, index) {
  var tags = tweet.hashtags;
  return Array.isArray(tags)
    && tags.indexOf('wstdays') !== -1;
}
```

# Операции с массивом. map

```
// Теперь в dtDdList массив с твитами в виде dt-dd-списка
var dtDdList = filteredTweets.map(render);

// Превращаем массив объектов твитов в HTML-строки
function render(tweet, index) {
  return '<dt>' + tweet.text + '</dt>' +
    '<dd>' + tweet.user + '</dd>' +
    '<dd>' + tweet.hashtags.join(', ') + '</dd>';
}
```

# Цепочки вызовов

```
// Теперь в result лежит HTML с деревом твитов
var result = '<dl>' + tweets.filter(isWstdaysHashtagExists)
                                .map(render)
                                .join('\n') + '</dl>';
```

```
// Выбираем только твиты с хештегом #wstdays
```

```
function isWstdaysHashtagExists(tweet, index) {
  var tags = tweet.hashtags;
  return Array.isArray(tags)
    && tags.indexOf('wstdays') !== -1;
}
```

```
// Превращаем массив объектов твитов в HTML-строки
```

```
function render(tweet, index) {
  return '<dt>' + tweet.text + '</dt>' +
    '<dd>' + tweet.user + '</dd>' +
    '<dd>' + tweet.hashtags.join(', ') + '</dd>';
}
```

# Операции с массивом. reduce

```
var likesCount = tweets.reduce(getTotalLikes, 0)

function getTotalLikes(acc, item) {
  return acc + item.likes;
}

likesCount; // 194
```



# Операции с массивом. reduce

```
var flattenTagsList = tweets.reduce(flattenHashtags, []);

function flattenHashtags(acc, item) {
  return acc.concat(item.hashtags || []);
}

flattenTagsList;
// ['wstdays', 'wstdays', 'mails', 'wstdays', 'wstdays',
// 'pokemongo', 'wstdays', 'wstdays', 'html', 'wstdays',
// 'nodejs', 'html', 'svg']
```

# Операции с массивом. reduce

```
var stat = flattenTagsList.reduce(getHashtagsStats, {});

function getHashtagsStats(acc, item) {
  if (!acc.hasOwnProperty(item)) {
    acc[item] = 0;
  }

  acc[item]++;

  return acc;
}

stat;
// { html: 2, mails: 1, nodejs: 1, pokemongo: 1,
//   svg: 1, wstdays: 7 }
```

# Цепочки вызовов

```
var hashtagsStat = tweets.reduce(flattenHashtags, [])  
                           .reduce(getHashtagsStats, {});
```

```
function flattenHashtags(acc, item) {  
  return acc.concat(item.hashtags || []);  
}
```

```
function getHashtagsStats(acc, item) {  
  if (!acc.hasOwnProperty(item)) {  
    acc[item] = 0;  
  }
```

```
  acc[item]++;
```

```
  return acc;  
}
```

```
hashtagsStat: // {html: 2, mail: 1, reddit: 1, pokemon: 1}
```

# Полезные функции для работы со строками

`toLowerCase` — приводим строку к нижнему регистру

`trim` — удаляет пробельные символы с обеих сторон

`startsWith` — начинается ли строка с подстроки

Все функции для работы со строками

# Полезные функции для работы с массивами

`sort` — сортируем массив

`every` — удовлетворяют ли **все элементы** массива условию

`some` — удовлетворяет ли **хотя бы один элемент** массива условию

`shift` — выталкивает из массива первый элемент и возвращает его

`unshift` — добавляет элемент в начало массива

Все функции для работы с массивами

Домашнее задание

Телефонная книга