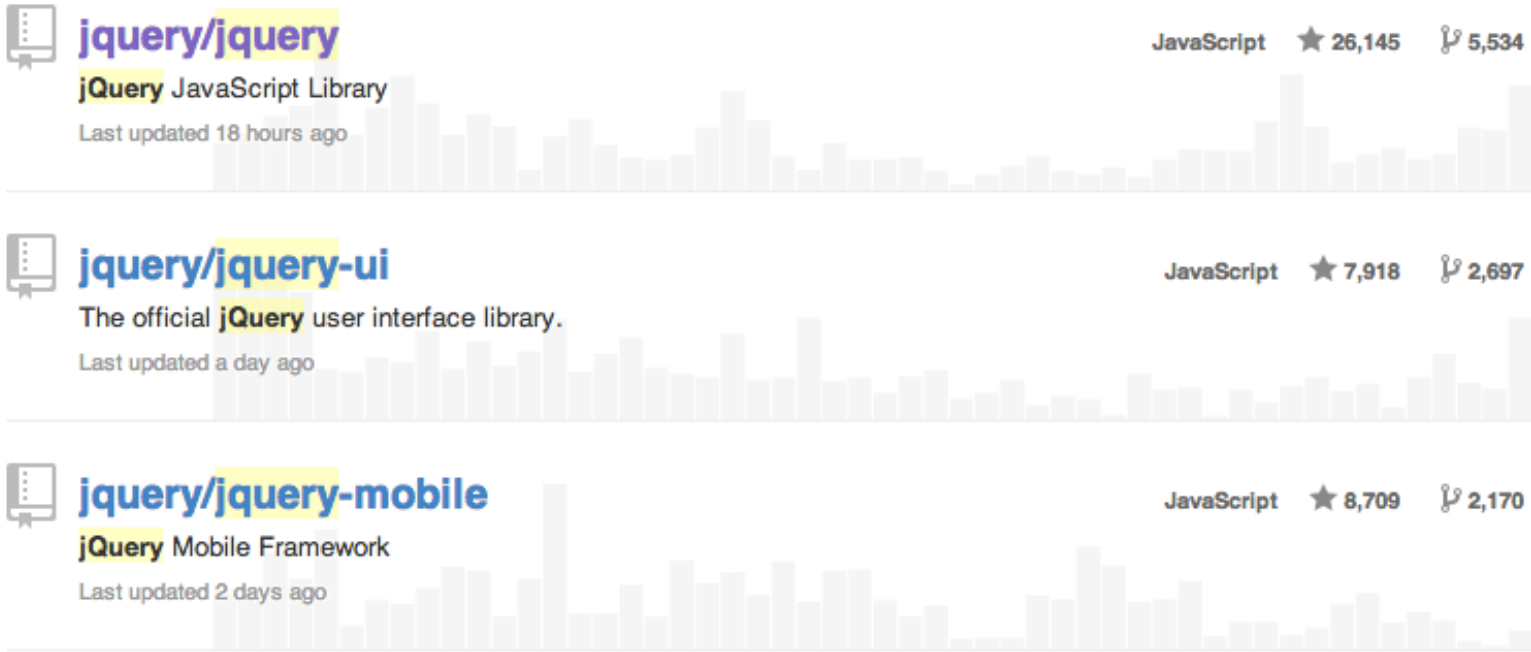
Javascript. Начало

@alexeyfrank

jQuery


jquery/jquery
jQuery JavaScript Library
Last updated 18 hours ago
JavaScript
26,145
5,534
jquery/jquery-ui
The official jQuery user interface library.
Last updated a day ago
JavaScript
7,918
2,697
jquery/jquery-mobile
jQuery Mobile Framework
Last updated 2 days ago
JavaScript
8,709
2,170

jQuery. Setup

```
1  <head>
2  <script type="text/javascript" src="jquery.js"></script>
3  </head>
```

Google. CDN

https://developers.google.com/speed/libraries/devguide?hl=ru

jQuery

snippet: `<script src="//ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"></script>`

site: http://jquery.com/

versions: 2.0.3, 2.0.2, 2.0.1, 2.0.0, 1.10.2, 1.10.1, 1.10.0, 1.9.1, 1.9.0, 1.8.3...

note: 1.2.5 and 1.2.4 are not hosted due to their short and unstable lives in the wild.

jQuery UI

snippet: `<script src="//ajax.googleapis.com/ajax/libs/jqueryui/1.10.3/jquery-ui.min.js"></script>`

site: http://jqueryui.com/

versions: 1.10.3, 1.10.2, 1.10.1, 1.10.0, 1.9.2, 1.9.1, 1.9.0, 1.8.24, 1.8.23, 1....

note: This library depends on jQuery. You must also load jQuery before loading this module. Version 1.8.3 is not hosted due to its short 1.8.4.

jQuery.domReady

```
1 $(document).ready(function() {
2   // Place your code here
3 });
4
5 // or
6
7 $(function() {
8   // Place your code here
9 });
```

jQuery.selectors

```
$('#sidebar');     // выбор элемента с id = sidebar
$('.post');        // выбор элементов с class = post
$('div#sidebar'); // выбор элемента div с id = sidebar
$('div.post');     // выбор элементов div с class = post
```

jQuery.selectors

```
1 $('div span');         // выбор всех span элементов в элементах div
2
3 $('div').find('span');// выбор всех span элементов в элементах div
4
5 $('div > span')  // выбор всех span элементов в div, где span является прямым потомком div'а
6
7 $('div, span')         // выбор всех div и span элементов
```

jQuery.selectors

```
$('span + img');        // выбор всех img элементов перед которыми идут span элементы
$('span ~ img');        // выбор всех img элементов после первого элемента span
$('#banner').prev();    // выбор предыдущего элемента от найденого
$('#banner').next();    // выбор следующего элемента от найденого
```

jQuery.selectors

```
$('div:first');     // выбираем первый div в доме
$('div:last');      // выбираем последний div в доме
$('div:not(.red)'); // выбираем div'ы у которых нету класса red
$('div:even');      // выбираем четные div'ы
$('div:odd');       // выбираем нечетные div'ы
$('div:eq(N)');     // выбираем div идущим под номером N в DOMe
$('div:gt(N)');     // выбираем div'ы, индекс которых больше чем N в DOMe
$('div:lt(N)');     // выбираем div'ы, индекс которых меньше чем N в DOMe
$(':header');       // выбо заголовоков h1, h2, h3 и т.д.
$('div:animated');  // выбор элементов с активной анимацией
```

jQuery.events

```
1 $('a').click(function(e) {
2   // Click logic here
3   e.preventDefault();
4 });
5
```

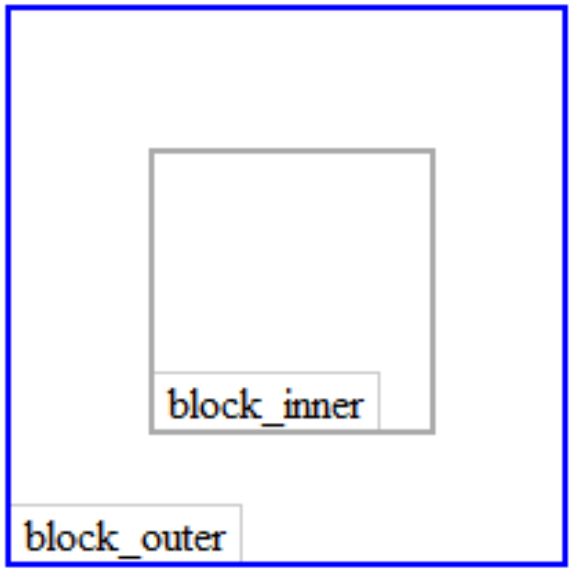
jQuery.events

```
1 $('input[type="text"]').blur(function() {
2   console.log(this); // typeof HTMLInputElement
3   var value = this.value.trim();
4   this.value = value;
5 });
6
```

jQuery.events

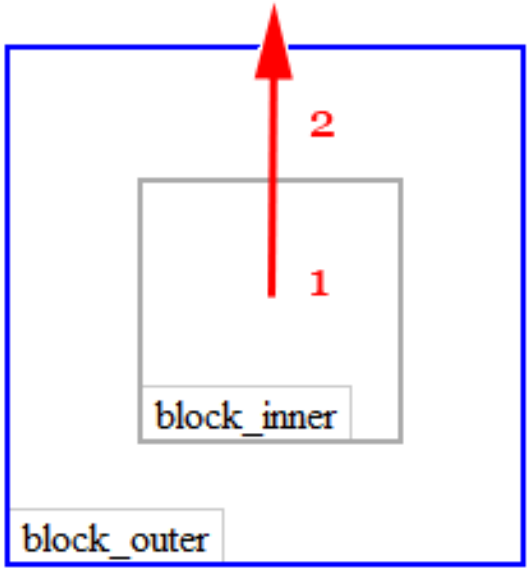
```
1 $('a').on('click', function() {
2   // Do some action here
3 });
4
5 // Or
6 $('#content').on('click', 'a', function() {
7   // Do some action here
8 });
```

jQuery. Event bubbling

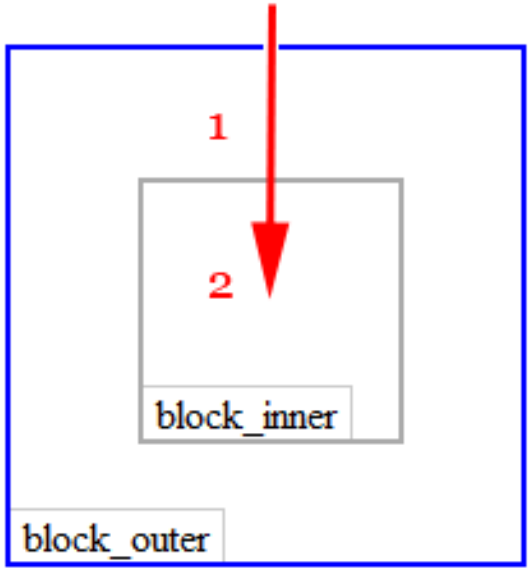



```
<div id="block_outer">
    <div id="block_inner"></div>
</div>
```

jQuery.events bubbling



BUBBLING



CAPTURING

jQuery.events bubbling

```
1 $('#block_outer').on('click', '#block_inner', function() {
2   // Do some action here
3 });
4 // ... remove and add new div#block_inner
5
6 $('#block_inner').click();
```

jQuery. Ajax

```
$.ajax({
    url: '/ajax/example.html',              // указываем URL и
    dataType : "json",                      // тип загружаемых данных
    success: function (data, textStatus) { // вешаем свой обработчик на функцию success
        $.each(data, function(i, val) {    // обрабатываем полученные данные
            /* ... */
        });
    }
});
```

jQuery. Ajax

```
$.post('ajax/example.json', {
    id: 1,
    name: 'Mary'
}, function(response){
    if (response.errors) {
      Notifier.error(response.errors);
    } else {
      Notifier.success();
    }
}, 'json');
```

jQuery. Effects

```
01  // вызов метода
02  $('#my').slideUp();
03
04  // аналогичен
05  $('#my').animate({height:0,padding:0}, function(){
06      $(this).css({display:"none"});
07  });
08
09  // вызов метода
10  $('#my').fadeOut();
11
12  // аналогичен
13  $('#my').animate({opacity:0}, function(){
14      $(this).css({display:"none"});
15  });
```

jQuery. Effects

```
01  // установит прозрачность элемента в ноль, прозрачность изменяется от 0 до 1
02  $('#my').animate({opacity:0});
03
04  // наращиваем высоту элемента на 200px
05  $('#my').animate({height:'+=200px'});
06
07  // уменьшаем ширину элемента на 50px
08  $('#my').animate({width:'-=50px'});
09
10  // наращиваем высоту элемента до 20in
11  $('#my').animate({height:'20in'});
```

jQuery. Проблемы

```
$.get('http://api.example.com/users/1', function(usersRes) {
    if (userRes.users) {
        $.get('http://insta.api.com/images', {users: usersRes.users}, function(imagesRes) {
            if (imagesRes) {
                var $ul = $('#images');
                imagesRes.images.forEach(function(image) {
                    var itemTpl = '<li>' + image.title + '</li>';
                    $ul.append(itemTpl);
                });
            } else {
                Notifier.notify( I18n.t('images_not_found') );
            }
        }, 'json');
    } else {
        Notifier.notify( I18n.t('images_not_found') );
    }
}, 'json');
```

Backbone.JS

- Наследование
- Модели (Models)
- Коллекции [Underscore JS]
- Представления (Views)
- Событийно-ориентированные коммуникации (Events)
- Персистентность (Sync)
- Маршрутизация и HTML5 pushState
- Потенциальная возможность тестирования (Jasmine/QUnit/SimonJS)

Backbone.Models

• Модель - это одна запись.

• Содержит интерактивные данные

• Реализует

– Конвертацию

– Валидацию

– Вычисление свойств

– Контроль доступа

• Реализуется посредством наследования стандартного класса Backbone.Model

• Базовый класс набор необходимых и вспомогательных методов для работы со свойствами (get/set, escape, has, toJSON, fetch, isValid, validate, save)

Backbone. Models

```
var Photo = Backbone.Model.extend({
    defaults: {
        src: placeholder.jpg,
        title:  an  image   placeholder,
        coordinates:    [0,0]
    },
    initialize: function()  {
        this.on("change:src", function()    {
            var src = this.get("src"); `
            console.log('Image source updated   to' + src);
        });
    },
    changeSrc:  function( source ) {
        this.set({ src: source });
    }
});
// Later
var somePhoto = new Photo({ src: "test.jpg", title:"testing"});
somePhoto.changeSrc("magic.jpg");
```

Backbone.Model Validations

```
1  var Photo = Backbone.Model.extend({
2      validate: function(attribs){
3          if(attribs.src  === undefined) {
4              return  "Remember to set a source for your image!";
5          }
6      },
7      initialize: function() {
8          console.log(this    model    has been    initialized);
9          this.on("error", function(model, error) {
10             console.log(error);
11         });
12     }
13 });
14 var myPhoto = new Photo();
15 myPhoto.set({ title: "On the beach" });
```

Backbone.Collections

• Позаимствованы из http://underscorejs.org/ – Унаследовали все методы

• Множество моделей, для которого реализованы:

– Фильтрация

– Сортировка

– Агрегация

• Реализуются посредством наследования стандартного класса Backbone.Collection

• Базовый класс также реализует полезные методы (get, set, length, push, pop, sort, create, fetch, pluck, where, ...)

• Позволяют централизовано работать с несколькими моделями

• Можно написать свой comparator

Backbone. Collections

```
1  var PhotoCollection = Backbone.Collection.extend({
2      model:  Photo
3  });
4  var a = new Photo({ title: 'my vacation'}),
5      b = new Photo({ title: 'my holiday'}),
6      c = new Photo({ title: 'my weekend'});
7  var photoCollection = new PhotoCollection([a,b]);
8  photoCollection.add(c);
9  photoCollection.remove([a,b]);
10 photoCollection.remove(c);
```

Backbone. Views

- Главная идея - собрать интерфейс из представлений, связанных с моделью
- Реиспользуемый элемент пользовательского интерфейса
- Не содержит вёрстку документа
- Часто ассоциирован с моделью
- Создается посредством наследования базового класса Backbone.View

Backbone.Views

```
var PhotoSearch = Backbone.View.extend({
  el: $('#results'),
  render: function(event) {
    var compiled_template = _.template( $("#results-template").html() );
    this.$el.html( compiled_template(this.model.toJSON()) );
    return this;
  },
  events: {
    "submit #searchForm": this.search,
    "click  .reset":      this.reset,
    "click  .advanced":   this.switchContext
  },
  search:            function(event) {},
  reset:             function(event) {},
  switchContext:     function(event) {}
});
```

Backbone. Router

• Мэпит ссылки (URL) на функции в js

– Адресация в веб-приложении

– Работа с историей в браузере

• на низком уровне: hashchange или HTML5 pushState

• После того как создали все роутеры нужно скомандовать start

Backbone.history.start(), чтобы история начала вестись

Backbone.Router

```
var AppRouter = Backbone.Router.extend({
  routes: {
    "posts/:id": "getPost",
    // <a href="http://example.com/#/posts/121">Example</a>
    "download/*path": "downloadFile",
    // <a href="http://example.com/#/download/user/images/hey.gif">Download</a>
    ":route/:action": "loadView",
    // <a href="http://example.com/#/dashboard/graph">Load Route/Action View</a>
  }
});

// Later
var appRouter = new AppRouter();

appRouter.on('route:getPost', function(id) {
  alert(id); // 121
});
appRouter.on('route:downloadFile', function(path) {
  alert(path); // user/images/hey.gif
});
appRouter.on('route:loadView', function(route, action) {
  alert(route + "_" + action); // dashboard_graph
});
```

Другие полезные библиотеки

1) momentjs
2) underscore / lodash
3) underscore.string
4) Three.js
5) Jasmine
6) Leaflet

momentjs

```
moment().subtract('days', 10).calendar();
moment().subtract('days', 6).calendar();
moment().subtract('days', 3).calendar();
moment().subtract('days', 1).calendar();
moment().calendar();
moment().add('days', 1).calendar();
moment().add('days', 3).calendar();
moment().add('days', 10).calendar();
```

```
11/08/2013
Last Tuesday at 1:35 PM
Last Friday at 1:35 PM
Yesterday at 1:35 PM
Today at 1:35 PM
Tomorrow at 1:35 PM
Thursday at 1:35 PM
11/28/2013
```

lodash

```
var sum = _.reduce([1, 2, 3], function(sum, num) {
  return sum + num;
});
// → 6
```

```
var characters = [
  { 'name': 'barney', 'age': 36, 'pets': ['hoppy'] },
  { 'name': 'fred',   'age': 40, 'pets': ['baby puss', 'dino'] }
];

_.where(characters, { 'age': 36 });
// → [{ 'name': 'barney', 'age': 36, 'pets': ['hoppy'] }]
```

underscore.string

```
_.swapCase('hELLO')
=> 'Hello'

_.include("foobar", "ob")
=> true

_('Hello world').count('l')
=> 3

_('my name is epeli').titleize()
=> 'My Name Is Epeli'

_('some_class_name').classify()
=> 'SomeClassName'
```

Three.js

```
var camera, scene, renderer;
var geometry, material, mesh;
function init() {
    camera = new THREE.PerspectiveCamera( 75,
                                          window.innerWidth / window.innerHeight,
                                          1, 10000 );
    camera.position.z = 1000;

    scene = new THREE.Scene();
    geometry = new THREE.CubeGeometry( 200, 200, 200 );
    material = new THREE.MeshBasicMaterial( { color: 0xff0000, wireframe: true } );

    mesh = new THREE.Mesh( geometry, material );
    scene.add( mesh );

    renderer = new THREE.CanvasRenderer();
    renderer.setSize( window.innerWidth, window.innerHeight );

    document.body.appendChild( renderer.domElement );
}
```

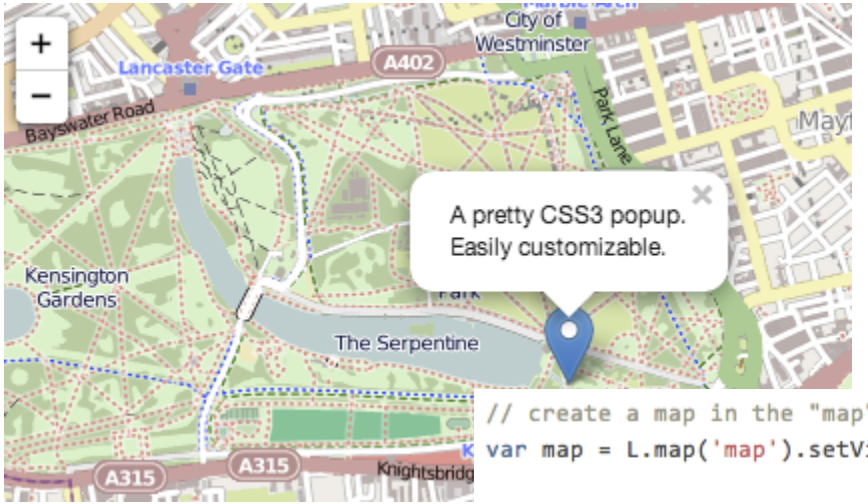
Jasmine

```
describe("A suite", function() {
  it("contains spec with an expectation", function() {
    expect(true).toBe(true);
  });
});
```

```
it("The 'toMatch' matcher is for regular expressions", function() {
  var message = 'foo bar baz';

  expect(message).toMatch(/bar/);
  expect(message).toMatch('bar');
  expect(message).not.toMatch(/quux/);
});
```

Leaflet




```
// create a map in the "map" div, set the view to a given place and zoom
var map = L.map('map').setView([51.505, -0.09], 13);

// add an OpenStreetMap tile layer
L.tileLayer('http://{s}.tile.osm.org/{z}/{x}/{y}.png', {
    attribution: '&copy; <a href="http://osm.org/copyright">OpenStreetMap</a> contributors'
}).addTo(map);

// add a marker in the given location, attach some popup content to it and open the popup
L.marker([51.5, -0.09]).addTo(map)
    .bindPopup('A pretty CSS3 popup. <br> Easily customizable.')
    .openPopup();
```