Примеры и исходный код EmberJS

Пример 1
http://jsbin.com/ifilet/3/edit

Пример 2
http://jsbin.com/elovel/3/edit

Пример 3
http://jsbin.com/aconeh/3/edit

Пример 4
http://jsbin.com/ijejap/1/edit

Пример 5
http://jsbin.com/ibabun/1/edit

Пример 6
http://jsbin.com/arebem/2/edit

Пример 7
http://jsbin.com/apuboh/3/edit

Пример 8
http://jsbin.com/apuboh/3/edit

212.26.132.30:5555-5565

Создание интерактивных приложений

ember JS

Angular JS

BACKBONE.JS

панель - левая

панель - правая

панель - верхная

панель - нижняя

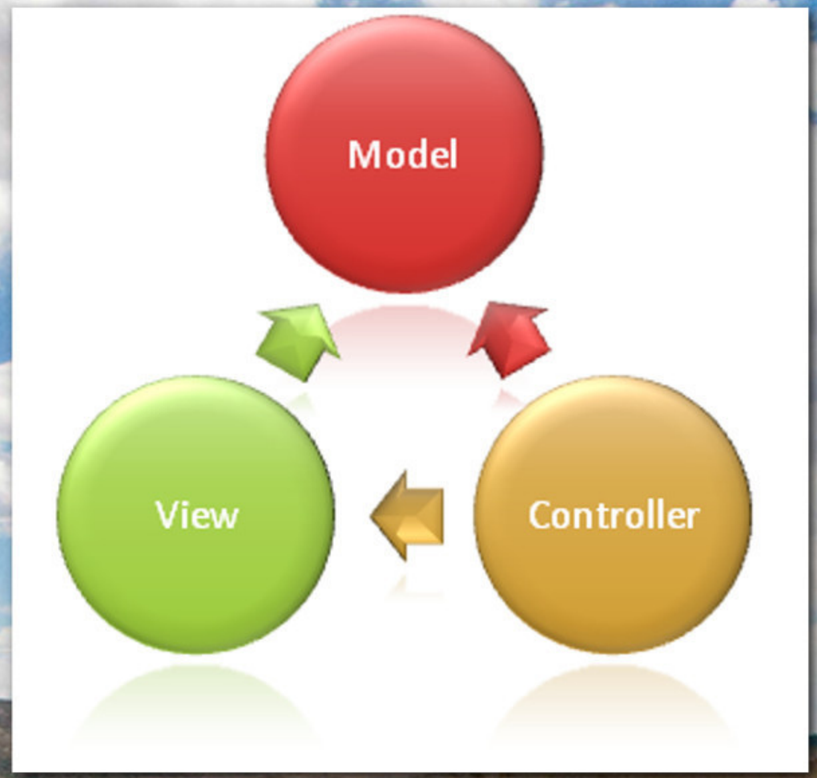
Авторизация

kog@adamant.ua lugatex@yahoo.com github.com/kapranov

Капранов Олег (Ruby on Rails)

Меню

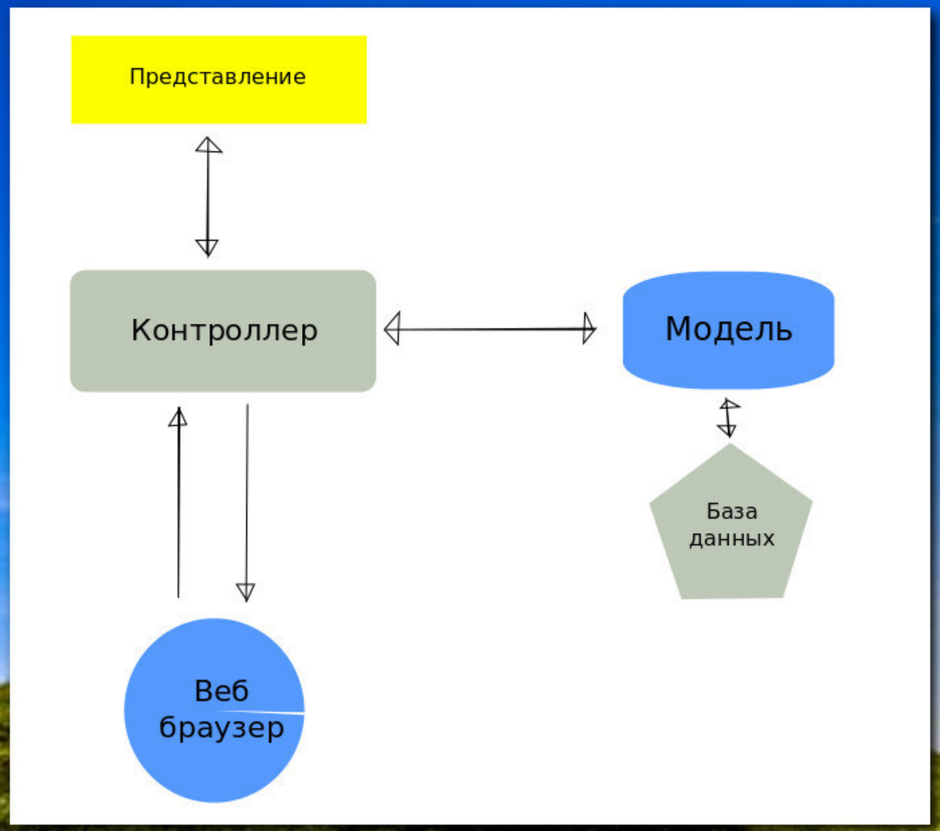
ADAMANT ОПЕРАТОР СИСТЕМНЫХ РЕШЕНИЙ

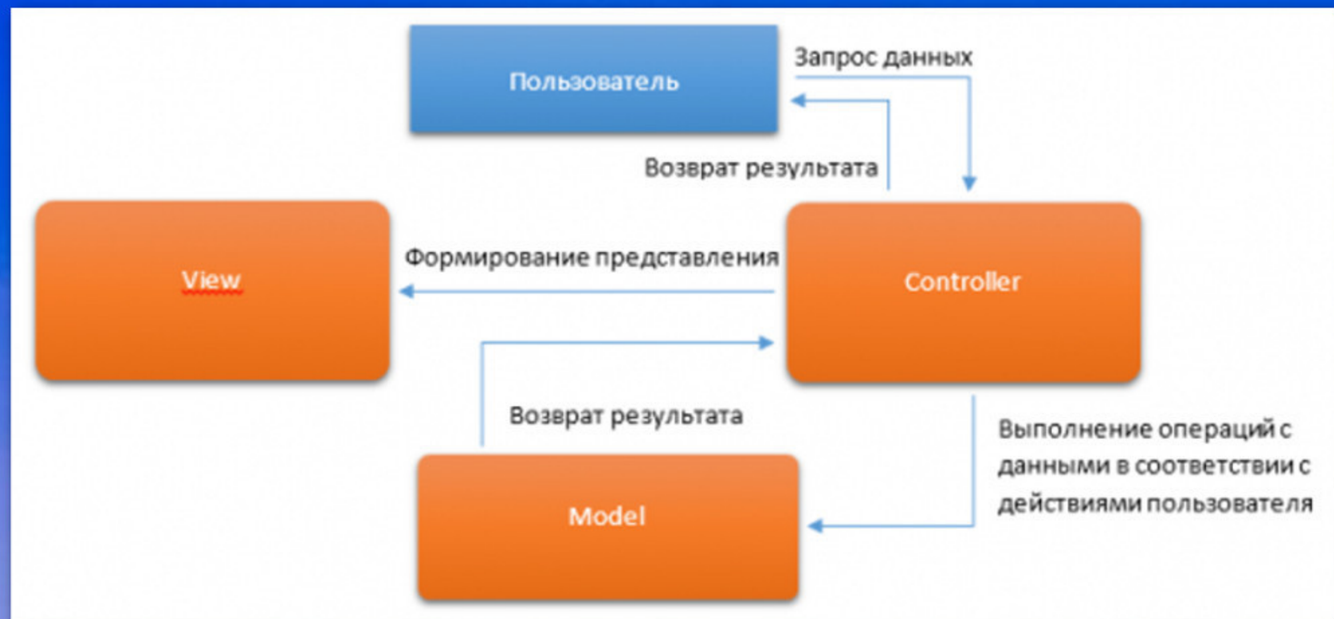


Model-View-Controller
«модель-представление-поведение»

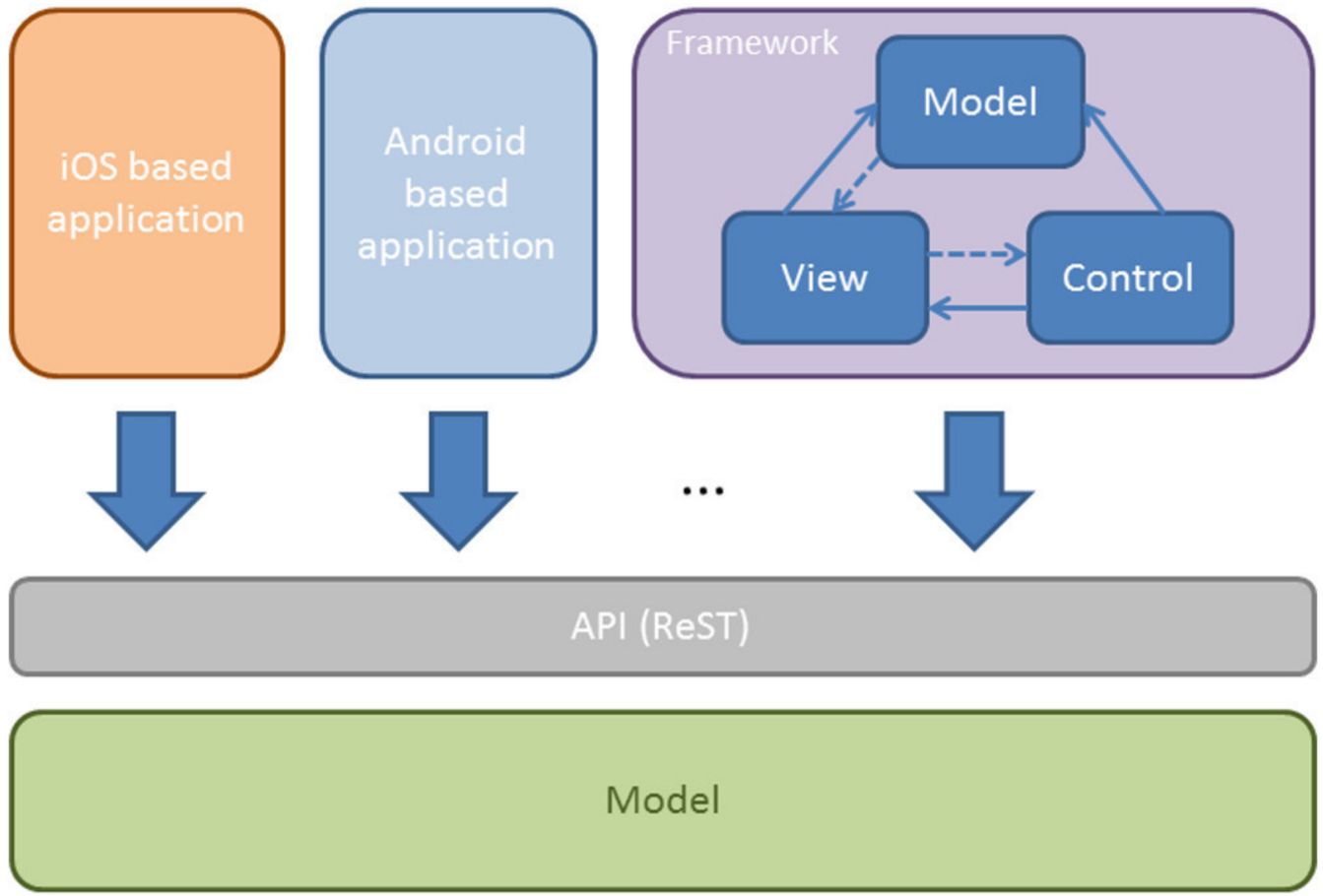
Model-view-controller
«модель-представление-поведение»
«модель-представление-контроллер»
«модель-вид-контроллер»

схема использования нескольких шаблонов проектирования, с помощью которых модель данных приложения, пользовательский интерфейс и взаимодействие с пользователем разделены на три отдельных компонента так, что модификация одного из компонентов оказывает минимальное воздействие на остальные.




Пользователь
Запрос данных
Возврат результата
View
Формирование представления
Controller
Возврат результата
Выполнение операций с данными в соответствии с действиями пользователя
Model

REST - передача состояния представления.



Данные в REST должны передаваться в виде небольшого количества стандартных форматов (HTML, XML, JSON).
Сетевой протокол (как и HTTP) должен поддерживать кэширование, не должен зависеть от сетевого слоя,
не должен сохранять информацию о состоянии между парами «запрос-ответ».
Такой подход обеспечивает масштабируемость системы и позволяет ей эволюционировать с новыми требованиями.

Functions

Functions are defined by an optional list of parameters in parentheses, an arrow, and the function body. The empty function looks like this: `->`

```
square = (x) -> x * x
cube   = (x) -> square(x) * x
```

load

```
var cube, square;

square = function(x) {
  return x * x;
};

cube = function(x) {
  return square(x) * x;
};
```

run: cube(5)

Functions may also have default values for arguments. Override the default value by passing a non-null argument.

```
fill = (container, liquid = "coffee") ->
  "Filling the #{container} with #{liquid}..."
```

load

```
var fill;

fill = function(container, liquid) {
  if (liquid == null) {
    liquid = "coffee";
  }
  return "Filling the " + container + " with " + liquid + "...";
};
```

run: fill("cup")

Objects and Arrays

The CoffeeScript literals for objects and arrays look very similar to their JavaScript cousins. When each property is listed on its own line, the commas are optional. Objects may be created using indentation instead of explicit braces, similar to YAML.

```
song = ["do", "re", "mi", "fa", "so"]

singers = {Jagger: "Rock", Elvis: "Roll"}

bitlist = [
  1, 0, 1
  0, 0, 1
  1, 1, 0
]

kids =
  brother:
    name: "Max"
    age:  11
  sister:
    name: "Ida"
    age:  9
```

load

```
var bitlist, kids, singers, song;

song = ["do", "re", "mi", "fa", "so"];

singers = {
  Jagger: "Rock",
  Elvis: "Roll"
};

bitlist = [1, 0, 1, 0, 0, 1, 1, 1, 0];

kids = {
  brother: {
    name: "Max",
    age: 11
  },
  sister: {
    name: "Ida",
    age: 9
  }
};
```

run: song.join(" ... ")

A framework for creating **ambitious** web applications

DOWNLOAD THE STARTER KIT

1.1.2: production (min + gzip 64kb) | debug | Handlebars

Follow @emberjs Follow on

MORE PRODUCTIVE OUT OF THE BOX.

Write dramatically less code with Ember's Handlebars integrated templates that update automatically when the underlying data changes.

Don't waste time making trivial choices. Ember.js incorporates common idioms so you can focus on what makes your app special, not reinventing the wheel.

Ember.js is built for productivity. Designed with developer ergonomics in mind, its friendly APIs help you get your job done—fast.

AUTO-UPDATING HANDLEBARS TEMPLATES

Ember makes Handlebars templates even better, by ensuring your HTML stays up-to-date when the underlying model changes. To get started, you don't even need to write any JavaScript.

application.hbs

```
<div>
```

PROJECTS

Ember
Ember Data

TAG: V1.1.2

MODULES

ember
ember-application
ember-debug
ember-handlebars
ember-handlebars-compiler
ember-metal
ember-routing
ember-runtime
ember-testing
ember-views

NAMESPACES

Ember
Ember.Instrumentation
Ember.Location
Ember.String
Ember.platform
Ember.run

CLASSES

Ember.ActionHandler
Ember.Application
Ember.Array
Ember.ArrayController
Ember.ArrayProxy
Ember.Binding
Ember.Checkbox
Ember.CollectionView
Ember.Comparable
Ember.Component
Ember.ComputedProperty
Ember.ContainerView
Ember.Controller

EMBER NAMESPACE

PA

DEFINED IN: packages/ember-metal/lib/core.js:8
MODULE: ember-metal

All Ember methods and functions are defined inside of this namespace. You generally should not add new properties to this namespace as it may be overwritten by future versions of Ember.

You can also use the shorthand `Em` instead of `Ember`.

Ember-Runtime is a framework that provides core functions for Ember including cross-platform functions, support for property observing and objects. Its focus is on small size and performance. You can use this in place of or along-side other cross-platform libraries such as jQuery.

The core Runtime framework is based on the jQuery API with a number of performance optimizations.

Index | Methods | Properties | Events

Show: ☑ Inherited ☐ Protected ☐ Private ☐ Deprecated

$

Defined in packages/ember-views/lib/core.js:9

Alias for jQuery

A Ember.NativeArray

Defined in packages/ember-runtime/lib/system/native_array.js:124

Creates an `Ember.NativeArray` from an Array like object. Does not modify the original object. Ember.A is not needed if `Ember.EXTEND_PROTOTYPES` is `true` (the default value). However, it is recommended that you use Ember.A when creating addons for ember or when you can not garentee that `Ember.EXTEND_PROTOTYPES` will be `true`.

Example

```js
var Pagination = Ember.CollectionView.extend({
  tagName: 'ul',
  classNames: ['pagination'],
  init: function() {
    this._super();
    if (!this.get('content')) {
      this.set('content', Ember.A([]));
    }
  }
});
```

Returns:

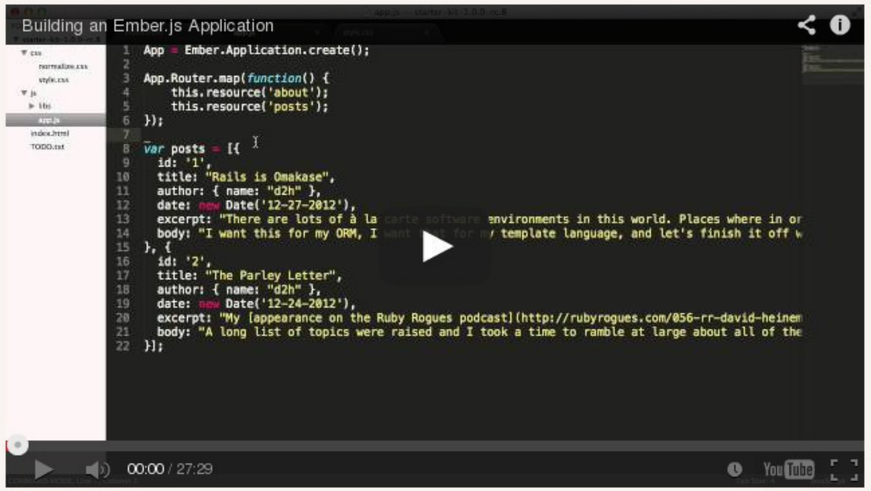
Ember.NativeArray

GETTING STARTED
CONCEPTS
THE OBJECT MODEL
APPLICATION
TEMPLATES
ROUTING
COMPONENTS
CONTROLLERS
MODELS
VIEWS
ENUMERABLES
TESTING
CONFIGURING EMBER.JS
COOKBOOK
UNDERSTANDING EMBER.JS
CONTRIBUTING TO EMBER.JS

EMBER.JS GUIDES

Welcome to the Ember.js guides! This documentation will take you from total beginner to Ember expert. It is designed to start from the basics, and slowly increase to more sophisticated concepts until you know everything there is to know about building awesome web applications.

To help you get started, we've also made a 30-minute screencast that will guide you through building a full-featured Ember.js application:



Source code for the app we build in the video is available at https://github.com/tildeio/bloggr-client

Most of these guides are designed to help you start building apps right away. If you'd like to know more about the thinking behind Ember.js, you'll find what you're looking for in the *Understanding Ember.js* section.

These guides are written in Markdown and are available on GitHub, inside the `source/guides` directory. If there is something missing, or you find a typo or mistake, please help us by filing an issue or submitting a pull request. Thanks!

We're excited for all of the great apps you're going to build with Ember.js. To get started, select a topic from the left. They are presented in the order that we think will be most useful to you as you're learning Ember.js, but you can also jump to whatever seems most interesting.

Good luck!

We're done with Ember.js Guides. Next up: Getting Started - Getting Started →

HTML enhanced for web apps!

View on GitHub

Download (1.2.1)

Follow +AngularJS on g+1 +55800 Follow @angularjs 25.9K followers Tweet 4,359

Why AngularJS?

HTML is great for declaring static documents, but it falters when we try to use it for declaring dynamic views in web-applications. AngularJS lets you extend HTML vocabulary for your application. The resulting environment is extraordinarily expressive, readable, and quick to develop.

Alternatives

Other frameworks deal with HTML's shortcomings by either abstracting away HTML, CSS, and/or JavaScript or by providing an imperative way for manipulating the DOM. Neither of these address the root problem that HTML was not designed for dynamic views.

Extensibility

AngularJS is a toolset for building the framework most suited to your application development. It is fully extensible and works well with other libraries. Every feature can be modified or replaced to suit your unique development workflow and feature needs. Read on to find out how.

The Basics

index.html

```
1.  <!doctype html>
2.  <html ng-app>
3.    <head>
4.      <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.2.0/
    angular.min.js"></script>
5.    </head>
6.    <body>
7.      <div>
8.        <label>Name:</label>
9.        <input type="text" ng-model="yourName" placeholder="Enter a name here">
10.       <hr>
11.       <h1>Hello {{yourName}}!</h1>
12.     </div>
```

Hint: hover over `me`.

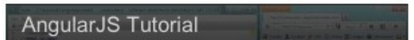
Edit Me

Name:

hello

Hello hello!

Watch as we build this app



API Reference

AngularJS API Docs

Improve this doc View source

Welcome to the AngularJS API docs page. These pages contain the AngularJS reference materials for version **1.2.1 underscore-empathy**.

The documentation is organized into **modules** which contain various components of an AngularJS application. These components are directives, services, filters, providers, types, global APIs and testing mocks.

Angular Namespaces $ and $$ To prevent accidental name collisions with your code, Angular prefixes names of public objects with $ and names of private objects with $$. Please do not use the $ or $$ prefix in your code.

Angular Namespace

ng (core module)

This module is provided by default and contains the core components of AngularJS.

Directives	This is the core collection of directives you would use in your template code to build an AngularJS application. Some examples include: `ngClick`, `ngInclude`, `ngRepeat`, etc...
Services / Factories	This is the core collection of services which are used within the DI of your application. Some examples include: `$compile`, `$http`, $routeParams, `$location`, etc...
Filters	The core filters available in the ng module are used to transform template data before it is renders within directives and expressions. Some examples include: `filter`, `date`, `currency`, `lowercase`, `uppercase`, etc...
Global APIs	The core global API functions are attached to the angular object. These core functions are useful for low level JavaScript operations within your application. Some examples include: `angular.copy()`, `angular.equals()`, `angular.element()`, etc...

ngRoute

Use ngRoute to enable URL routing to your application. The ngRoute module supports URL management via both hashbang and HTML5 pushState.

Include the **angular-route.js** file and set **ngRoute** as a dependency for this to work in your application.

Services / Factories	The following services are used for route management: • $routeParams is used to access the querystring values present in the URL. • $route is used to access the details of the route that is currently being accessed. • $routeProvider is used to register routes for the application.
Directives	The ngView directive will display the template of the current route within the page.

ngAnimate

Use ngAnimate to enable animation features into your application. Various core ng directives will provide animation hooks into your application when ngAnimate is included. Animations are defined by using CSS transitions/animations or JavaScript callbacks.

Include the **angular-animate.js** file and set **ngAnimate** as a dependency for this to work in your application.

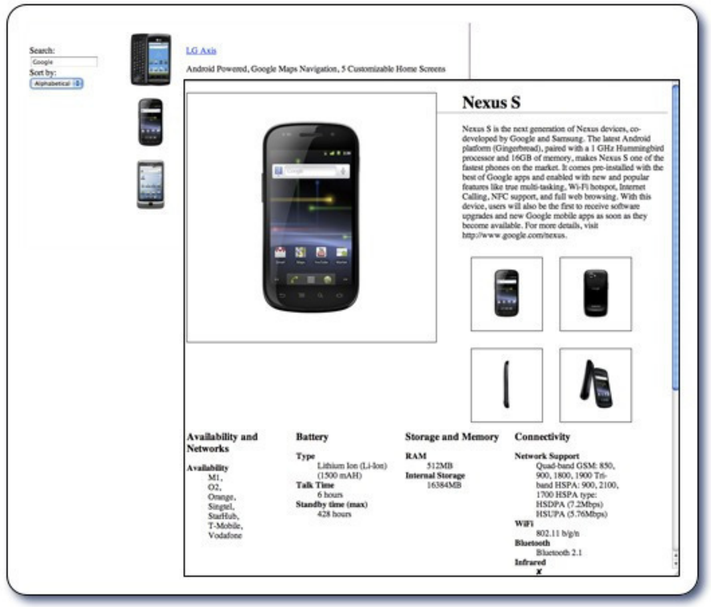
AngularJS - v1.2.2-751f05

0 - Bootstrapping
1 - Static Template
2 - Angular Templates
3 - Filtering Repeaters
4 - Two-way Data Binding
5 - XHRs & Dependency Injection
6 - Templating Links & Images
7 - Routing & Multiple Views
8 - More Templating
9 - Filters
10 - Event Handlers
11 - REST and Custom Services
12 - Applying Animations
The End

Tutorial

Improve this doc

A great way to get introduced to AngularJS is to work through this tutorial, which walks you through the construction of an AngularJS web app. The app you will build is a catalog that displays a list of Android devices, lets you filter the list to see only devices that interest you, and then view details for any device.



Work through the tutorial to see how Angular makes browsers smarter — without the use of extensions or plug-ins. As you work through the tutorial, you will:

- See examples of how to use client-side data binding and dependency injection to build dynamic views of data that change immediately in response to user actions.
- See how Angular creates listeners on your data without the need for DOM manipulation.
- Learn a better, easier way to test your web apps.
- Learn how to use Angular services to make common web tasks, such as getting data into your app, easier.

And all of this works in any browser without modification to the browser!

When you finish the tutorial you will be able to:

- Create a dynamic application that works in any browser.
- Define the differences between Angular and common JavaScript frameworks.
- Understand how data binding works in AngularJS.
- Use the angular-seed project to quickly boot-strap your own projects.
- Create and run tests.
- Identify resources for learning more about AngularJS.

The tutorial guides you through the entire process of building a simple application, including writing and running unit and end-to-end tests. Experiments at the end of each step provide suggestions for you to learn more about AngularJS and the application you are building.

You can go through the whole tutorial in a couple of hours or you may want to spend a pleasant day really digging into it. If you're looking for a shorter introduction to AngularJS, check out the Getting Started document.

Working with the code

You can follow this tutorial and hack on the code in either the Mac/Linux or the Windows environment. The tutorial relies

Backbone.js (1.1.0)
» GitHub Repository
» Annotated Source

Introduction

Upgrading

Events
- on
- off
- trigger
- once
- listenTo
- stopListening
- listenToOnce
- **Catalog of Built-in Events**

Model
- extend
- constructor / initialize
- get
- set
- escape
- has
- unset
- clear
- id
- idAttribute
- cid
- attributes
- changed
- defaults
- toJSON
- sync
- fetch
- save
- destroy
- **Underscore Methods (6)**
- validate
- validationError
- isValid
- url
- urlRoot
- parse
- clone
- isNew
- hasChanged
- changedAttributes
- previous
- previousAttributes

Collection
- extend
- model
- constructor / initialize
- models
- toJSON
- sync
- **Underscore Methods (28)**
- add
- remove
- reset
- set
- get
- at
- push

BACKBONE.JS

Backbone.js gives structure to web applications by providing **models** with key-value binding and custom events, **collections** with a rich API of enumerable functions, **views** with declarative event handling, and connects it all to your existing API over a RESTful JSON interface.

The project is hosted on GitHub, and the annotated source code is available, as well as an online test suite, an example application, a list of tutorials and a long list of real-world projects that use Backbone. Backbone is available for use under the MIT software license.

You can report bugs and discuss features on the GitHub issues page, on Freenode IRC in the `#documentcloud` channel, post questions to the Google Group, add pages to the wiki or send tweets to @documentcloud.

Backbone is an open-source component of DocumentCloud.

Downloads & Dependencies (Right-click, and use "Save As")

Development Version (1.1.0) *59kb, Full source, tons of comments*

Production Version (1.1.0) *6.4kb, Packed and gzipped* (Source Map)

Edge Version (master) *Unreleased, use at your own risk* build passing

Backbone's only hard dependency is Underscore.js (>= 1.5.0). For RESTful persistence, history support via Backbone.Router and DOM manipulation with Backbone.View, include **jQuery**, and **json2.js** for older Internet Explorer support. *(Mimics of the Underscore and jQuery APIs, such as Lo-Dash and Zepto, will also tend to work, with varying degrees of compatibility.)*

Introduction

When working on a web application that involves a lot of JavaScript, one of the first things you learn is to stop tying your data to the DOM. It's all too easy to create JavaScript applications that end up as tangled piles of jQuery selectors and callbacks, all trying frantically to keep data in sync between the HTML UI, your JavaScript logic, and the database on your server. For rich client-side applications, a more structured approach is often helpful.

With Backbone, you represent your data as Models, which can be created, validated, destroyed, and saved to the server. Whenever a UI action causes an attribute of a model to change, the model triggers a *"change"* event; all the Views that display the

- setElement
- attributes
- $ (jQuery)
- template
- render
- remove
- delegateEvents
- undelegateEvents

Utility

- Backbone.noConflict
- Backbone.$

Examples

- Todos
- DocumentCloud
- USA Today
- Rdio
- Hulu
- Quartz
- Gawker Media
- Flow
- Gilt Groupe
- NewsBlur
- WordPress.com
- Foursquare
- Bitbucket
- Disqus
- Delicious
- Khan Academy
- eLife Lens
- Do
- IRCCloud
- Pitchfork
- Spin
- ZocDoc
- Walmart Mobile
- Groupon Now!
- Basecamp
- Slavery Footprint
- Stripe
- Airbnb
- Diaspora
- SoundCloud Mobile
- Art.sy
- Pandora
- Inkling
- Code School
- CloudApp
- SeatGeek
- Easel
- Jolicloud
- Syllabus
- Salon.io
- TileMill
- Blossom
- Trello
- Tzigla

F.A.Q.

- Why Backbone?
- More Than One Way To Do It
- Nested Models & Collections
- Loading Bootstrapped Models
- Extending Backbone
- Traditional MVC
- Binding "this"
- Working with Rails

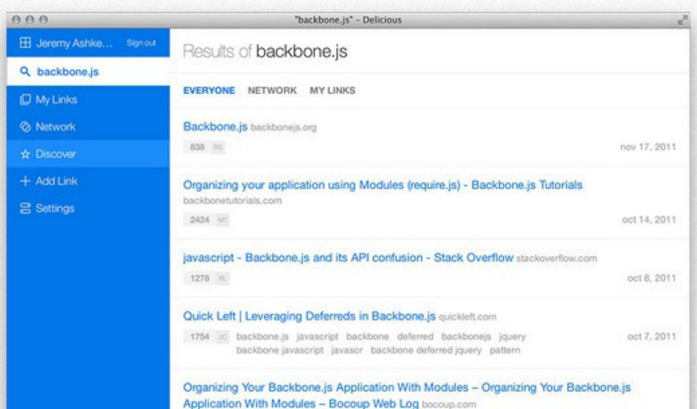
Change Log

Disqus

Disqus chose Backbone.js to power the latest version of their commenting widget. Backbone's small footprint and easy extensibility made it the right choice for Disqus' distributed web application, which is hosted entirely inside an iframe and served on thousands of large web properties, including IGN, Wired, CNN, MLB, and more.

Delicious

Delicious is a social bookmarking platform making it easy to save, sort, and store bookmarks from across the web. Delicious uses Chaplin.js, Backbone.js and AppCache to build a full-featured MVC web app. The use of Backbone helped the website and mobile apps share a single API service, and the reuse of the model tier made it significantly easier to share code during the recent Delicious redesign.



- setElement
- attributes
- $ (jQuery)
- template
- render
- remove
- delegateEvents
- undelegateEvents

Utility

- Backbone.noConflict
- Backbone.$

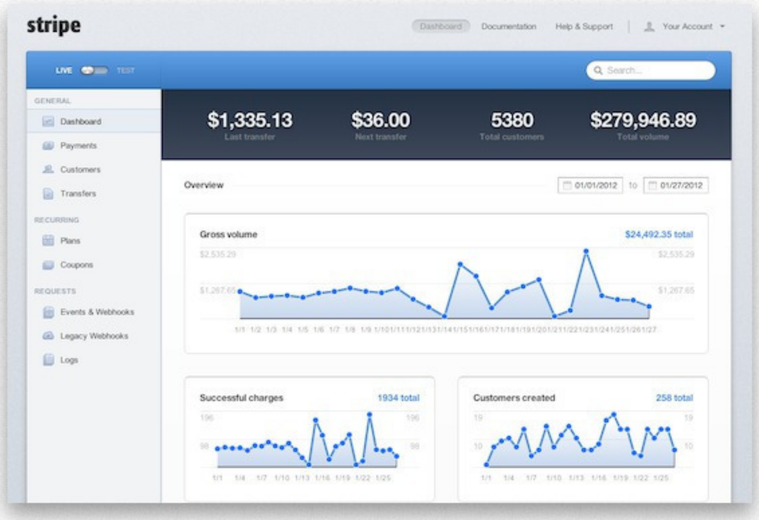
Examples

- Todos
- DocumentCloud
- USA Today
- Rdio
- Hulu
- Quartz
- Gawker Media
- Flow
- Gilt Groupe
- NewsBlur
- WordPress.com
- Foursquare
- Bitbucket
- Disqus
- Delicious
- Khan Academy
- eLife Lens
- Do
- IRCCloud
- Pitchfork
- Spin
- ZocDoc
- Walmart Mobile
- Groupon Now!
- Basecamp
- Slavery Footprint
- Stripe
- Airbnb
- Diaspora
- SoundCloud Mobile
- Art.sy
- Pandora
- Inkling
- Code School
- CloudApp
- SeatGeek
- Easel
- Jolicloud
- Syllabus
- Salon.io
- TileMill
- Blossom
- Trello
- Tzigla

F.A.Q.

- Why Backbone?
- More Than One Way To Do It
- Nested Models & Collections
- Loading Bootstrapped Models
- Extending Backbone
- Traditional MVC
- Binding "this"
- Working with Rails

Change Log



Airbnb

Airbnb uses Backbone in many of its products. It started with Airbnb Mobile Web (built in six weeks by a team of three) and has since grown to Wish Lists, Match, Search, Communities, Payments, and Internal Tools.

SoundCloud Mobile

SoundCloud is the leading sound sharing platform on the internet, and Backbone.js

CSS with superpowers

Sass is the most mature, stable, and powerful professional grade CSS extension language in the world.

Current Release: Media Mark (3.2.12) **Release Notes** **Fork on Github**

CSS COMPATIBLE

Sass is completely compatible with all versions of CSS. We take this compatibility seriously, so that you can seamlessly use any available CSS libraries.

FEATURE RICH

Sass boasts more features and abilities than any other CSS extension language out there. The Sass Core Team has worked endlessly to not only keep up, but stay ahead.

MATURE

Sass has been actively supported for almost 7 years by its loving Core Team.

INDUSTRY APPROVED

Over and over again, the industry is choosing Sass as the premier CSS extension language.

LARGE COMMUNITY

Sass is actively supported and developed by a consortium of several tech companies and hundreds of developers.

FRAMEWORKS

There are endless number of frameworks built with Sass. Compass, Bourbon, and Susy just to name a few.

Sass © 2006–2013 Hampton Catlin, Nathan Weizenbaum, Chris Eppstein, and numerous contributors.
It is available for use and modification under the MIT License.

Fork me on GitHub

The dynamic stylesheet language.

LESS extends CSS with dynamic behavior such as variables, mixins, operations and functions.

LESS runs on both the server-side (with Node.js and Rhino) or client-side (modern browsers only).

Download less.js

version 1.5.0
changelog

Write some LESS:

```
@base: #f938ab;

.box-shadow(@style, @c) when (iscolor(@c)) {
  box-shadow:         @style @c;
  -webkit-box-shadow: @style @c;
  -moz-box-shadow:    @style @c;
}
.box-shadow(@style, @alpha: 50%) when (isnumber(@alpha)) {
  .box-shadow(@style, rgba(0, 0, 0, @alpha));
}
.box {
  color: saturate(@base, 5%);
  border-color: lighten(@base, 30%);
  div { .box-shadow(0 0 5px, 30%) }
}
```

Compile to CSS:

```
npm install -g less
lessc styles.less styles.css
```


Variables

Variables allow you to specify widely used values in a single place, and then re-use them throughout the style sheet, making global changes as easy as changing one line of code.

```
// LESS

@color: #4D926F;

#header {
  color: @color;
}
h2 {
  color: @color;
}
```

```
/* Compiled CSS */

#header {
  color: #4D926F;
}
h2 {
  color: #4D926F;
}
```

Mixins

Mixins allow you to embed all the properties of a class into another class by simply including the class name as one of its properties. It's just like variables, but for whole classes. Mixins can also behave like functions, and take arguments, as seen in the example below.

```
// LESS

.rounded-corners (@radius: 5px) {
  -webkit-border-radius: @radius;
```

```
/* Compiled CSS */

#header {
  -webkit-border-radius: 5px;
```

compass

Compass is an open-source *CSS Authoring Framework*.

Why designers love Compass.

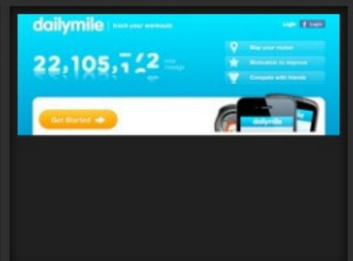
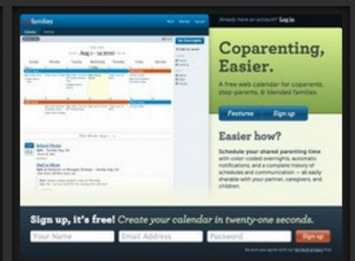
1. Experience cleaner markup without presentational classes.
2. It's chock full of the web's best reusable patterns.
3. It makes creating sprites a breeze.
4. Compass mixins make CSS3 easy.
5. Create beautiful typographic rhythms.
6. Download and create extensions with ease.

Compass uses Sass.

Sass is an extension of CSS3 which adds nested rules, variables, mixins, selector inheritance, and more. Sass generates well formatted CSS and makes your stylesheets easier to organize and maintain.

ONLINE JS CONFERENCE
Online, live JavaScript Summit 2013 - Nov. 19-21
Powered by Fusion

Brilliant people use Compass, including these *wildly talented* folks:

Transferring data from compass-style.org...

Bourbon

GITHUB DOCS

A simple and lightweight mixin library for Sass.

```
ol {
  > li {
    background-color: #F4F5F7;
    @include background-image( linear-gradient(to left, #FFF, #F4F5F7) );
    float: left;
    font-family: $lucida-grande;
    position: relative;
    @include transition( all 0.8s );

    &::after {
      content: " ";
      height: 23px;
      @include position( absolute, 10px -13px 0 0 );
      @include transform( rotate(45deg) skew(20deg, 30deg) );
      width: 23px;
    }
  }
}
```

CSS Output

```
-webkit-transform: rotate(45deg) skew(20deg, 30deg);
   -moz-transform: rotate(45deg) skew(20deg, 30deg);
    -ms-transform: rotate(45deg) skew(20deg, 30deg);
     -o-transform: rotate(45deg) skew(20deg, 30deg);
        transform: rotate(45deg) skew(20deg, 30deg);
```

Forget about vendor prefixes.

The mixins contain up-to-date vendor prefixes for support in modern browsers.

```
// Bourbon mixin with Sass

section {
  @include linear-gradient(to top, red, orange);
}
```

```
/* Compiled CSS  */

section {
  background-color: red;
  background-image: -webkit-linear-gradient(bottom, red, orange);
  background-image:         linear-gradient(to top, red, orange);
}
```

Write CSS faster and easier.

The library contains mixins, functions, and other addons to build kick ass web stuff.

Bootstrap

Sleek, intuitive, and powerful mobile first front-end framework for faster and easier web development.

Download Bootstrap Download source

ADS VIA Carbon
Reach designers who are up to date. Advertise with Carbon Ads.
ads via Carbon

Star 61,139 Fork 21,582 Follow @twbootstrap 105K follo Tweet 3,203

Currently v3.0.2 Bootstrap 2.3.2 docs GitHub project Examples Expo Blog

Foundation

The most advanced responsive front-end framework in the world.

Download Foundation 4

★ 14.1k stargazers @foundationzurb

Latest Update on Github NOV 13, 2013 Commit 8b3427c43800ed301e2cd005f0a72545cd465709 »

Contribute Now

What's new in 4?

Responsive design gets lighter, faster and more advanced, so you can **code smarter**.

Mobile First

Now you can build for small devices first. Then, as devices get larger and larger, layer in more complexity.

Semantic

Everything is now semantic. Now you can have the cleanest markup without sacrificing the utility and speed of Foundation.

Awesome Javascript

All of the JS plugins have been rewritten to make them better, stronger, faster.

Shiny New HTML Templates

We've created some common layouts for you to start from so you can cook your site faster.

Go to Templates

YAML 4 CSS Framework

A modular CSS framework for truly *flexible*, *accessible* and *responsive* websites

Quick Start Demos

Here are some examples that show the YAML CSS framework in full flight.

Flexible Grid

A flexible and responsive grid layout showing the basic usage of YAMLs grid elements.

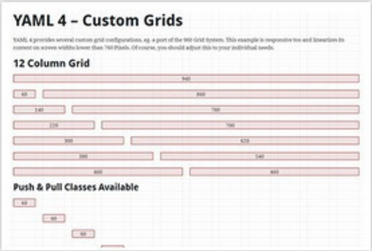
View Example

Columns & Grids

A responsive layout with a flexible content column, some nested grid elements and a fixed width sidebar.

View Example

Custom Grid (960px)



A fixed grid layout, based on 960 Grid System. It shows the use of custom grid configurations.

View Example

RTL Support

A demo for YAMLs support of Hebrew or Arabic languages, that have «right to left» text flow.

View Example

Introduction

About

YAML is continuously developed since 2005. It's strongly focussed on device independence and accessibilty.

It provides provides layout bullet-proof modules for flexible layouts.

File Structure

Within the download package fo you will find a folder called *yaml* with the following file stucture. This folder contains all components of the framework and should stay unchanged for easy updating.

If you want to modify any file, put a copy in your local CSS folder and reference it from there

Pure

A set of small, responsive CSS modules that you can use in every web project.

```
<link rel="stylesheet" href="http://yui.yahooapis.com/pure/0.3.0/pure-min.css">
```

Get Started | View on GitHub

Base	Grids	Forms	Buttons	Tables	Menus
1.2KB	0.9KB	1.4KB	0.8KB	0.5KB	1.2KB

CSS with a minimal footprint.

Pure is ridiculously tiny. The entire set of modules clocks in at **4.4KB* minified and gzipped**, without forgoing responsive styles, design, or ease of use. Crafted with mobile devices in mind, it was important to us to keep our file sizes small, and every line of CSS was carefully considered. If you decide to only use a subset of these modules, you'll save even more bytes.

* We can add correctly :) the numbers above are individual module sizes; when grouped together gzipping compresses them even more.

Your CSS foundation.

Pure builds on Normalize.css and provides layout and styling for native HTML elements, plus the most common UI components. It's what you need, without the cruft.

Made with mobile in mind.

Pure is responsive out of the box, so elements look great on all screen sizes. Customize your styles with the Skin Builder to make your style your own.

Stays out of your way.

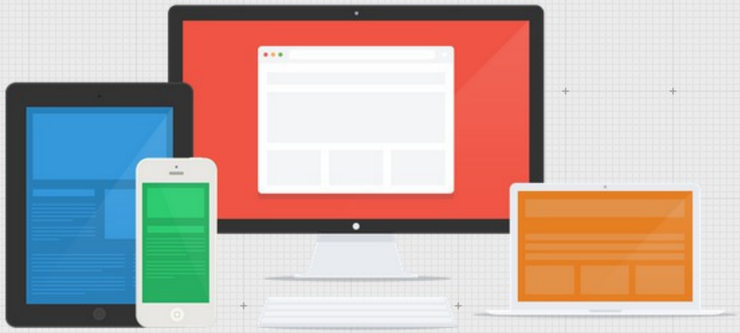
Pure has minimal styles and encourages you to write your application styles on top of it. It's designed to get out of your way and makes it easy to override styles.

BOOTSTRAP HERO

Tour Bad Ass Resources Pricing Contact

LET YOUR CUSTOMERS SPEND THEIR TIME CUSTOMIZING YOUR APP.

A system for letting users customize the look and style of THEIR pages within YOUR service.



START NOW

Want in the Beta?
Please enter your Email.

Your Email Address

Get Notified

What SAAS Developers are Saying

Andrew Culver
Founder LimeLightApp, Koudoku

Ryan Castillo
Consultant / Software Developer

Ian Taylor
Founder CarryNorfolk, OurCareDirect

A lightweight templating engine

slim-template/slim

gem version 2.0.2 | build passing
dependencies up-to-date | code climate 3.6
Flattr 3 | Gittip

Fork me on Github!

```
doctype html
html
  head
    title Slim Examples
    meta name="keywords" content="template language"
    meta name="author" content=author
    javascript:
      alert('Slim supports embedded javascript!')

  body
    h1 Markup examples

    #content
      p This example shows you how a basic Slim file looks like.

      == yield

      - unless items.empty?
        table
          - for item in items do
            tr
              td.name = item.name
              td.price = item.price
      - else
        p
          | No items found.  Please add some inventory.
          Thank you!

  div id="footer"
    = render 'footer'
    | Copyright © #{year} #{author}
```

Emblem.js

a concise, beautiful, and fully compatible templating alternative for Handlebars.js

Syntax

Idioms

Installation

About

Embercast

Live Demo

Star 529 Fork 24


```
p Introducing Emblem.js: a new templating language
  that compiles to Handlebars.js

ul
  li Indentation-based (like Slim, Jade, HAML, etc.)
  li Compiles to Handlebars; full compatibility with
     both custom and built-in Handlebars helpers
  li Ember.js-friendly syntax and full compatibility
     with auto-updating functionality.

/ Emblem is HTML5-aware; Render HTML if a line starts with
  known HTML tag, else invoke HB helper or perform property lookup.
  These defaults are easily overridable when necessary.
ul
  each person
    li = name

section.ember-features
  / easy, natural Ember bindAttr support
  h1 class=foo The class name of this element is bound to `foo`
  p class=isActive:flashing:inactive Ember.js bound css classes

  / implicit Ember action handlers
  a click="doIt" href="#" Hello, how are you doing, #{name}?

/ condense nested content into a single line if that's how you roll
#footer: ul.menu-items: each menu_items: li: a.menu-link href=url: link_text

if something
  p something was true!
else
  p something was false!
```


Fork me on GitHub

Emblem.js

Syntax

Tags

You can wrap content within an html5 element by just writing the element name at the beginning of a line.

This is supported for all valid html5 tags (list here.)

To use a non-html5 tag, just start the line with `%`.

EMBLEM

```
h1 Welcome to Emblem

%blink buy it on Kozmo.com
```

HANDLEBARS

```
<h1>Welcome to Emblem</h1>

<blink>buy it on Kozmo.com</blink>
```

Indentation Wrapping

To wrap one element in another, just indent and place the element below.

EMBLEM

```
footer
  ul
    li Hello
    li Goodbye
```

HANDLEBARS

```
<footer>
  <ul>
    <li>Hello</li>
    <li>Goodbye</li>
  </ul>
</footer>
```

CSS Classes

CSS classes can added to elements by using a (`.`) with the class name afterwards. You can place this after an element name (`div` is the default tag name).

You can chain multiple class names.

EMBLEM

```
.title Title

h1.logo Law Blog

button.btn.btn-large Submit
```

HANDLEBARS

```
<div class="title">Title</div>

<h1 class="logo">Law Blog</h1>

<button class="btn btn-large">Submit</button>
```

IDs

Element IDs can be added to elements by using `#` before the element id. Like CSS classes, you can provide an element name (`div` is the default tag name).

EMBLEM

```
#page-content Content

span#name Bob Lablah
```

HANDLEBARS

```
<div id="page-content">Content</div>
```

Примеры и исходный код EmberJS

Пример 1
http://jsbin.com/ifilet/3/edit

Пример 2
http://jsbin.com/elovel/3/edit

Пример 3
http://jsbin.com/aconeh/3/edit

Пример 4
http://jsbin.com/ijejap/1/edit

Пример 5
http://jsbin.com/ibabun/1/edit

Пример 6
http://jsbin.com/arebem/2/edit

Пример 7
http://jsbin.com/apuboh/3/edit

Пример 8
http://jsbin.com/apuboh/3/edit

212.26.132.30:5555-5565

Bins ▾ Add library Share HTML CSS JavaScript Console Output Log in Register Help

HTML ▾

```html
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-1.9.1.js"></script>
<script src="//cdnjs.cloudflare.com/ajax/libs/handlebars.js/1.0.0-rc.3/handlebars.js"></script>
<script src="//cdnjs.cloudflare.com/ajax/libs/ember.js/1.0.0-rc.1/ember.js">
</script>
<meta charset=utf-8 />
<title></title>
</head>
<body>

  <script type="text/x-handlebars">
    <h1>{{App.name}}</h1>
    <h2>{{App.secondsOnPage}}</h2>
  </script>

</body>
</html>
```

JavaScript ▾

```javascript
var App = Ember.Application.create()

App.name = "Пример 1 - Таймер";
App.secondsOnPage = 0;

setInterval(function() {
  App.set('secondsOnPage', App.get('secondsOnPage') + 1);
}, 1000);
```

1 warning

Output

Run with JS Auto-run JS ☑

Пример 1 - Таймер

14

Bins ▾ Add library Share HTML CSS JavaScript Console Output Log in Register Help

HTML ▾

```html
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-1.9.1.js"></script>
<script src="//cdnjs.cloudflare.com/ajax/libs/handlebars.js/1.0.0-rc.3/handlebars.js"></script>
<script src="//cdnjs.cloudflare.com/ajax/libs/ember.js/1.0.0-rc.1/ember.js">
</script>
<meta charset=utf-8 />
<title></title>
</head>
<body>

  <script type="text/x-handlebars">
    <h1>{{name}}</h1>
    <h2>{{timer}}</h2>
  </script>

</body>
</html>
```

JavaScript ▾

```javascript
var App = Ember.Application.create();

App.ApplicationRoute = Ember.Route.extend({

  model: function() {
    return { name: 'Пример 2 - маршрут', timer: 0 };
  },

  activate: function() {
    this.interval = setInterval(function() {
      var timer = this.get('controller.model.timer');
      this.set('controller.model.timer', timer + 1);
    }.bind(this), 1000);
  },

  deactivate: function() {
    clearInterval(this.interval);
  }

});
```

Output Run with JS Auto-run JS ☑ ↗

Пример 2 - маршрут

16

Bins ▾ Add library Share HTML CSS JavaScript Console Output Log in Register Help

HTML ▾

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery.js"></script>
<script src="//cdnjs.cloudflare.com/ajax/libs/handlebars.js/1.0.0-
rc.3/handlebars.js"></script>
  <script src="//ryanflorence.com/ember.min.js"></script>
<meta charset=utf-8 />
<link href="//netdna.bootstrapcdn.com/twitter-bootstrap/2.3.1/css/bootstrap-
combined.min.css" rel="stylesheet">
<title></title>
</head>
<body>

  <script type="text/x-handlebars">
    <ul class="nav well">
      {{#each model}}
        {{render "user" this}}
      {{/each}}
    </ul>
  </script>

  <script type="text/x-handlebars" id="user">
  <li>
    <p>
      <img {{bindAttr src="avatar"}}>
      <b>{{first}} {{last}}</b>
    </p>
  </li>
  </script>


</body>
</html>
```

JavaScript ▾

```
var App = Ember.Application.create();

App.ApplicationRoute = Ember.Route.extend({
  model: function() {
    return users;
  }
});

var users = [
  {
    id: 1,
    first: 'Андрей',
    last: 'Линде',
    avatar: 'https://si0.twimg.com/profile_images/3123276865
/5c069e64eb7f8e971d36a4540ed7cab2.jpeg'
  },
  {
    id: 2,
    first: 'Юрий',
    last: 'Арлинский',
    avatar: 'https://si0.twimg.com/profile_images/1317834118/avatar.png'
  },
  {
    id: 3,
    first: 'Якуда',
    last: 'Катц',
    avatar: 'https://si0.twimg.com/profile_images/3250074047
/46d910af94e25187832cb4a3bc84b2b5.jpeg'
  }
];
```

Output Run with JS Auto-run JS ☑

Андрей Линде

Юрий Арлинский

Якуда Катц

Bins ▾ Add library Share HTML CSS JavaScript Console Output Log in Register Help

HTML ▾

```html
<!DOCTYPE html>
<html>
<head>
  <meta name="description" content="Ember: Routes, {{outlet}}, and
{{#linkTo}}" />
<script src="http://code.jquery.com/jquery-1.9.1.js"></script>
<script src="//cdnjs.cloudflare.com/ajax/libs/handlebars.js/1.0.0-
rc.3/handlebars.js"></script>
<script src="//cdnjs.cloudflare.com/ajax/libs/ember.js/1.0.0-rc.1/ember.js">
</script>
<meta charset=utf-8 />
<link href="//netdna.bootstrapcdn.com/twitter-bootstrap/2.3.1/css/bootstrap-
combined.min.css" rel="stylesheet">
<title></title>
</head>
<body>

  <script type="text/x-handlebars">
    <div class="container">
      <div class="row-fluid">
        <ul class="nav nav-list span3 well">
          {{#each model}}
          <li>{{#linkTo "user" this}}{{first}}{{/linkTo}}</li>
          {{/each}}
        </ul>
        <div class="span9 well">{{outlet}}</div>
      </div>
    </div>
  </script>

  <script type="text/x-handlebars" id="user">
    <h2>
      {{first}} {{last}}
      <img {{bindAttr src="avatar"}} class="pull-right" width=50 />
    </h2>
    <dl>
      <dt>First</dt>
      <dd>{{first}}</dd>
      <dt>Last</dt>
      <dd>{{last}}</dd>
    </dl>
    </script>

</body>
</html>
```

JavaScript ▾

```javascript
var App = Ember.Application.create();

App.Router.map(function() {
  this.resource('user', {path: '/users/:user_id'});
});

App.UserRoute = Ember.Route.extend({
  model: function(params) {
    return users[params.user_id];
  }
});

App.ApplicationRoute = Ember.Route.extend({
  model: function() {
    return users;
  }
});

var users = [
  {
    id: 0,
    first: 'Андрей',
    last: 'Линде',
    avatar: 'https://si0.twimg.com/profile_images/3123276865
/5c069e64eb7f8e971d36a4540ed7cab2.jpeg'
  },
  {
    id: 1,
    first: 'Юрий',
    last: 'Арлинский',
    avatar: 'https://si0.twimg.com/profile_images/1317834118/avatar.png'
  },
  {
    id: 2,
    first: 'Якуда',
    last: 'Катц',
    avatar: 'https://si0.twimg.com/profile_images/3250074047
/46d910af94e25187832cb4a3bc84b2b5.jpeg'
  }
];
```

Output Run with JS Auto-run JS ☐

Андрей
Юрий
Якуда

Bins ▾ Add library Share HTML CSS JavaScript Console Output Log in Register Help

HTML ▾

```html
<!DOCTYPE html>
<html>
<head>
  <meta name="description" content="Ember: Routes, {{outlet}}, and
{{#linkTo}}" />
<script src="http://code.jquery.com/jquery-1.9.1.js"></script>
<script src="//cdnjs.cloudflare.com/ajax/libs/handlebars.js/1.0.0-
rc.3/handlebars.js"></script>
<script src="http://ember101.com/jsbin/ember.rc3.js"></script>
<meta charset=utf-8 />
<link href="//netdna.bootstrapcdn.com/twitter-bootstrap/2.3.1/css/bootstrap-
combined.min.css" rel="stylesheet">
<title></title>
</head>
<body>

  <script type="text/x-handlebars">
    <div class="container">
      <div class="row-fluid">
        <ul class="nav nav-list span3 well">
          {{#each model}}
          <li>{{#linkTo "user" this}}{{first}}{{/linkTo}}</li>
          {{/each}}
        </ul>
        <div class="span9 well">{{outlet}}</div>
      </div>
    </div>
  </script>

  <script type="text/x-handlebars" id="user">
    <h2>
      {{first}} {{last}}
      <img {{bindAttr src="avatar"}} class="pull-right" width=50 />
    </h2>
    <dl>
      <dt>First</dt>
      <dd>{{first}}</dd>
      <dt>Last</dt>
      <dd>{{last}}</dd>
    </dl>
    {{#linkTo "editUser" this class="btn btn-primary"}}Edit{{/linkTo}}
  </script>

  <script type="text/x-handlebars" id="editUser">
    <h2>Edit User</h2>
    <form class="form-inline" {{action save on="submit"}}>
      <label>
        First:
        {{input type="text" value=first}}
      </label><br>

      <label>
        Last:
        {{input type="text" value=last}}
      </label>

      <div class="controls">
        <button type="submit" class="btn btn-primary">Done</button>
      </div>
    </form>
  </script>

</body>
</html>
```

JavaScript ▾

```javascript
var App = Ember.Application.create();

App.Router.map(function() {
  this.resource('user', {path: '/users/:user_id'});
  this.resource('editUser', {path: '/users/:user_id/edit'});
});

App.ApplicationRoute = Ember.Route.extend({
  model: function() {
    return users;
  }
});

App.UserRoute = Ember.Route.extend({
  model: function(params) {
    return users[params.user_id];
  }
});

App.EditUserRoute = Ember.Route.extend({
  model: function(params) {
    return users[params.user_id];
  },

  events: {
    save: function() {
      var user = this.modelFor('editUser');
      this.transitionTo('user', user);
    }
  }
});




var users = [
  {
    id: 0,
    first: 'Андрей',
    last: 'Линде',
    avatar: 'https://si0.twimg.com/profile_images/3123276865
/5c069e64eb7f8e971d36a4540ed7cab2.jpeg'
  },
  {
    id: 1,
    first: 'Юрий',
    last: 'Арлинский',
    avatar: 'https://si0.twimg.com/profile_images/1317834118/avatar.png'
  },
  {
    id: 2,
    first: 'Якуда',
    last: 'Кац',
    avatar: 'https://si0.twimg.com/profile_images/3250074047
/46d910af94e25187832cb4a3bc84b2b5.jpeg'
  }
];
```

Output

Run with JS Auto-run JS

Андрей

Юрий

Якуда

Bins ▾ Add library Share HTML CSS JavaScript Console Output Log in Register Help

HTML ▾

```html
<!DOCTYPE html>
<html>
<head>
  <meta name="description" content="Ember: Routes, {{outlet}}, and
{{#linkTo}}" />
<script src="http://code.jquery.com/jquery-1.9.1.js"></script>
<script src="//cdnjs.cloudflare.com/ajax/libs/handlebars.js/1.0.0-
rc.3/handlebars.js"></script>
<script src="http://ember101.com/jsbin/ember.rc3.js"></script>
<meta charset=utf-8 />
<link href="//netdna.bootstrapcdn.com/twitter-bootstrap/2.3.1/css/bootstrap-
combined.min.css" rel="stylesheet">
<title></title>
</head>
<body>

  <script type="text/x-handlebars">
    <div class="container">
      <div class="row-fluid">
        <ul class="nav nav-list span3 well">
          <li><a href="#" {{action createUser}}>+ New User</a></li>
          {{#each model}}
          <li>{{#linkTo "user" this}}{{first}}{{/linkTo}}</li>
          {{/each}}
        </ul>
        <div class="span9 well">{{outlet}}</div>
      </div>
    </div>
  </script>

  <script type="text/x-handlebars" id="user">
    <h2>
      {{first}} {{last}}
      <img {{bindAttr src="avatar"}} class="pull-right" width=50 />
    </h2>
    <dl>
      <dt>First</dt>
      <dd>{{first}}</dd>
      <dt>Last</dt>
      <dd>{{last}}</dd>
    </dl>
    {{#linkTo "editUser" this class="btn btn-primary"}}Edit{{/linkTo}}
  </script>

  <script type="text/x-handlebars" id="editUser">
    <h2>Edit User</h2>
    <form class="form-inline" {{action save on="submit"}}>
      <label>
        First:
        {{input type="text" value=first}}
      </label><br>

      <label>
        Last:
        {{input type="text" value=last}}
      </label>

      <div class="controls">
        <button type="submit" class="btn btn-primary">Done</button>
      </div>
    </form>
  </script>
</body>
</html>
```

JavaScript ▾

```javascript
var App = Ember.Application.create();

App.Router.map(function() {
  this.resource('user', {path: '/users/:user_id'});
  this.resource('editUser', {path: '/users/:user_id/edit'});
});

App.ApplicationRoute = Ember.Route.extend({
  model: function() {
    return users;
  },

  events: {
    createUser: function() {
      var users = this.modelFor('application');
      var user = users.pushObject({
        id: users.length
      });
      this.transitionTo('editUser', user);
    }
  }
});

App.UserRoute = Ember.Route.extend({
  model: function(params) {
    return users[params.user_id];
  }
});

App.EditUserRoute = Ember.Route.extend({
  model: function(params) {
    return users[params.user_id];
  },

  events: {
    save: function() {
      var user = this.modelFor('editUser');
      this.transitionTo('user', user);
    }
  }
});


var users = [
  {
    id: 0,
    first: 'Андрей',
    last: 'Линде',
    avatar: 'https://si0.twimg.com/profile_images/3123276865/5c069e64eb7f8e971d36a4540ed7cab2.jpeg'
  },
  {
    id: 1,
    first: 'Юрий',
    last: 'Арлинский',
    avatar: 'https://si0.twimg.com/profile_images/1317834118/avatar.png'
  },
  {
    id: 2,
    first: 'Якуда',
    last: 'Катц',
    avatar: 'https://si0.twimg.com/profile_images/3250074047/46d910af94e25187832cb4a3bc84b2b5.jpeg'
  }
];
```

Output Run with JS Auto-run JS ☐

+ New User
Андрей
Юрий
Якуда

Bins ▾ Add library Share HTML CSS JavaScript Console Output Log in Register Help

HTML ▾

```html
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-1.9.1.js"></script>
<script src="//cdnjs.cloudflare.com/ajax/libs/handlebars.js/1.0.0-
rc.3/handlebars.js"></script>
<script src="//cdnjs.cloudflare.com/ajax/libs/ember.js/1.0.0-rc.2/ember.js">
</script>
<meta charset=utf-8 />
<title>JS Bin</title>
</head>
<body>

<script type="text/x-handlebars">
<h1>Начала списка</h1>
<ul>
  <li>{{#linkTo "about"}}о нас{{/linkTo}}</li>
  <li>{{#linkTo "login"}}логин{{/linkTo}}</li>
</ul>
{{outlet}}
</script>

<script type="text/x-handlebars" id="about">
<h2>О нас</h2>
<ul>
  <li>
    {{#linkTo "about.location"}}Location{{/linkTo}}
  </li>
  <li>
    {{#linkTo "about.product"}}Product{{/linkTo}}
  </li>
</ul>
{{outlet}}
</script>


<script type="text/x-handlebars" id="index">
<h2>Индекс</h2>
</script>

<script type="text/x-handlebars" id="login">
<h2>Логин</h2>
</script>

<script type="text/x-handlebars" id="about/index">
<h3>Индекс</h3>
</script>


<script type="text/x-handlebars" id="about/location">
<h3>Расположение</h3>
</script>

<script type="text/x-handlebars" id="about/product">
<h3>Товар</h3>
</script>

</body>
</html>
```

JavaScript ▾

```javascript
var App = Ember.Application.create();

App.Router.map(function() {
  this.resource('about', function() {
    this.route('product');
    this.route('location');
  });
  this.resource('login');
});
```

Output Run with JS Auto-run JS ☐ ↗

Начала списка

- о нас
- логин

Индекс

Bins ▾ Add library Share HTML CSS JavaScript Console Output Log in Register Help

HTML ▾

```
<!DOCTYPE html>
<html>
<head>
<meta charset=utf-8 />
<title></title>
  <script src="http://ajax.googleapis.com/ajax/libs/jquery/2.0.2/
jquery.min.js"></script>
  <script src="http://cdnjs.cloudflare.com/ajax/libs/
handlebars.js/1.0.0/handlebars.js"></script>
  <script src="http://builds.emberjs.com/canary/ember.prod.js"></
script>

</head>
<body>
  <script type="text/x-handlebars">
    <h1>Пример</h1>
    {{outlet}}
  </script>

  <script type="text/x-handlebars" id="index">
    {{#link-to 'profile' 'арлинский'}}профиль арлинского{{/link-
to}}
  </script>

  <script type="text/x-handlebars" id="profile">
    <h2>Профиль пользователя</h2>

    Привет ваша имя: {{username}}
  </script>

</body>
</html>
```

JavaScript ▾

```
App = Ember.Application.create();

App.Router.map(function() {
  this.resource('profile', { path: '/profile/:username' });
});

App.ProfileRoute = Ember.Route.extend({
  model: function(params, transition) {
    return { username: params.username };
  },

  serialize: function(model) {
    return { username: model.get('username') };
  }
});
```

Output Run with JS Auto-run JS ☐ ↗

Пример

профиль арлинского