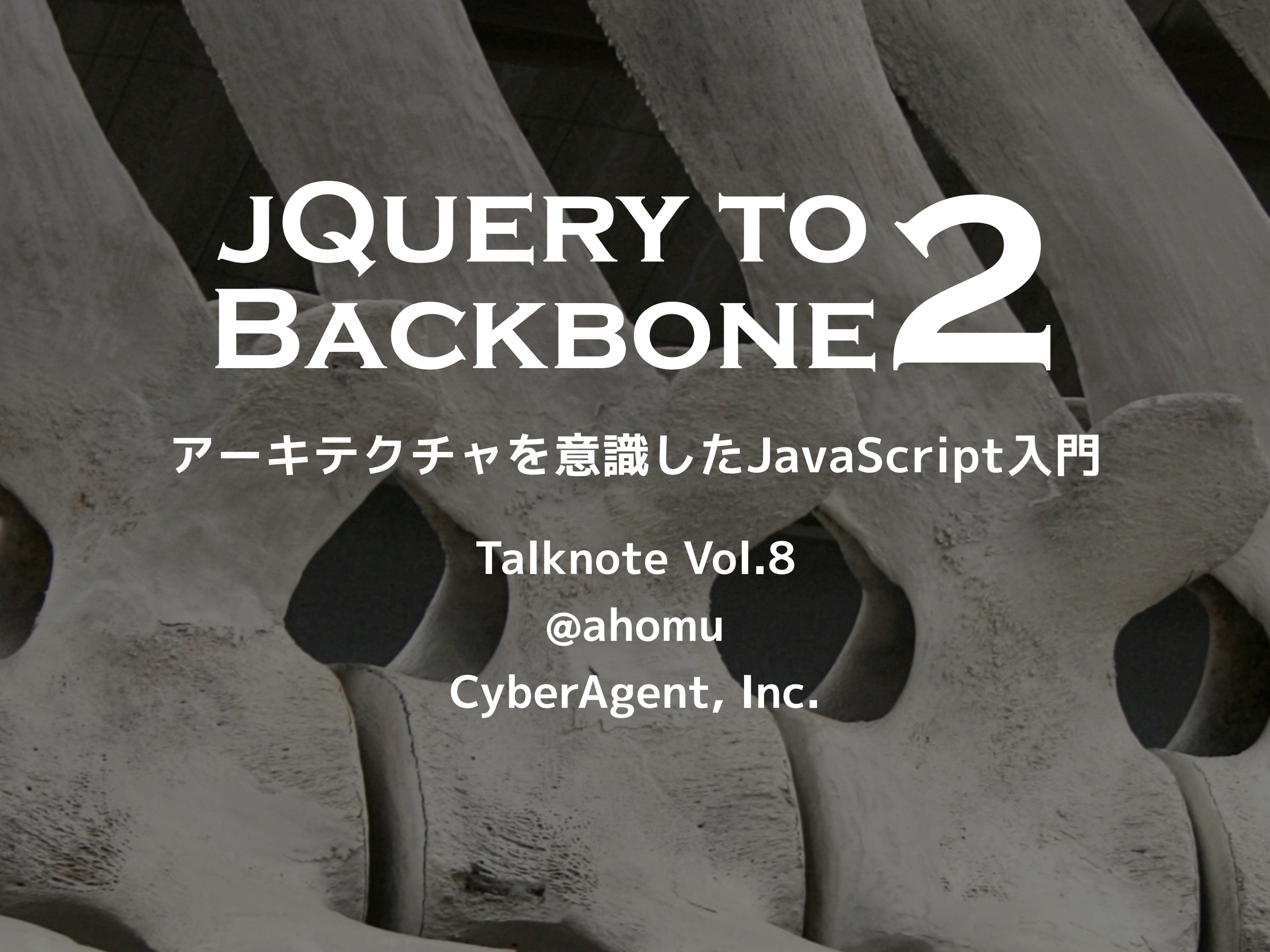


jQuery to Backbone 2

アーキテクチャを意識したJavaScript入門

Talknote Vol.8

@ahomu

CyberAgent, Inc.

佐藤 歩

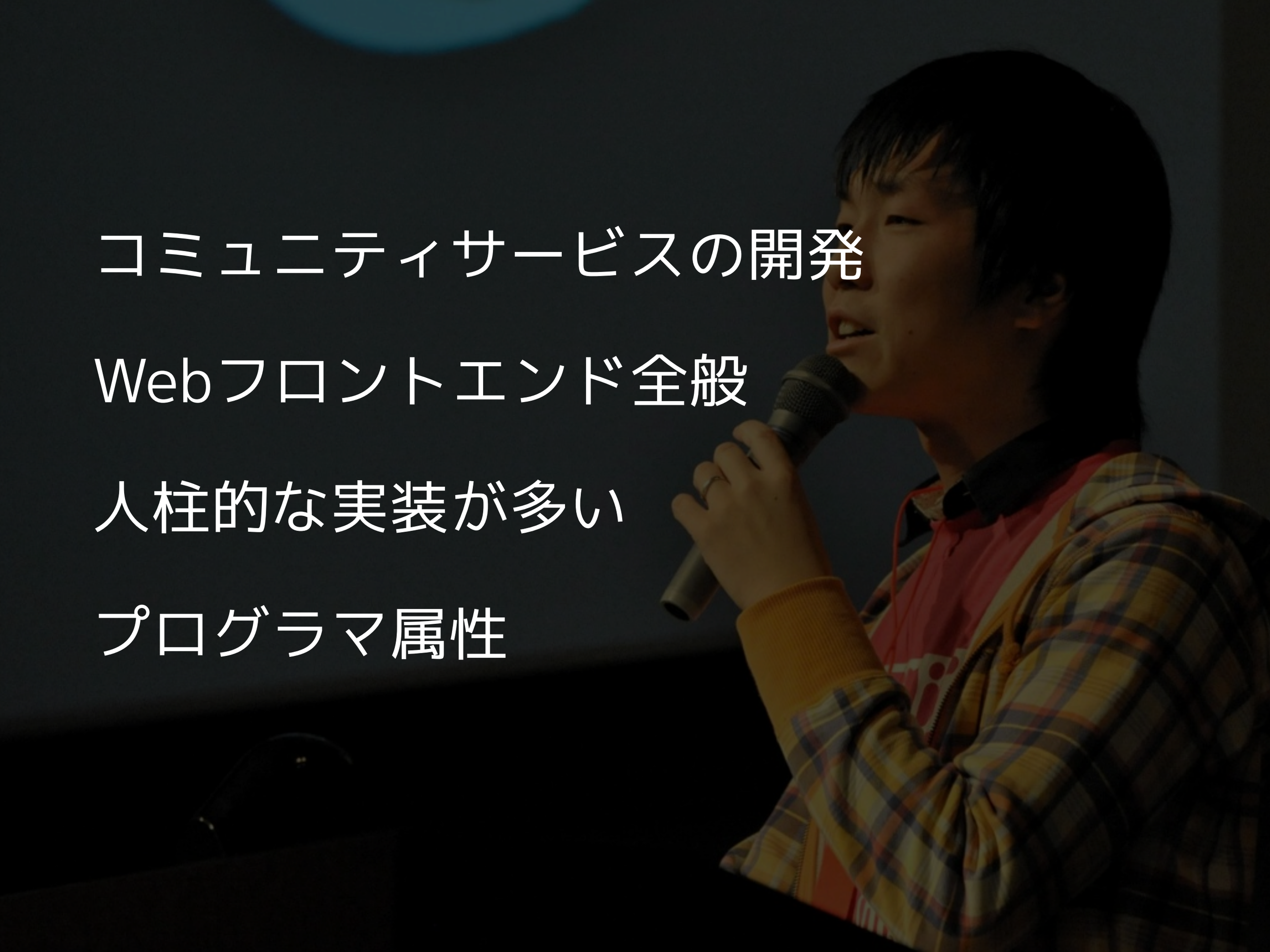
@ahomu

2012年8月以前

名古屋でWebプログラマ（PHP・JSなど）

2012年9月〜　**new!**

CAでフロントエンド専業にシフト



コミュニティサービスの開発

Webフロントエンド全般

人柱的な実装が多い

プログラマ属性

詳しくは

http://aho.mu

流れ

1. はじめに
2. jQueryについて
3. Backboneについて
4. jQuery to Backbone
5. まとめ

はじめに

jQueryは無くなりません

散らかったコードを減らそう

ライブラリとノウハウへの関心

jQueryについて
特徴と役割の振り返り

Plugins | Contribute | Events | Support | jQuery Foundation

jQuery
write less, do more.

Download | API Documentation | Blog | Plugins | Browser Support | Search jQuery

Lightweight Footprint

Only 32kB minified and gzipped. Can also be included as an AMD module

CSS3 Compliant

to find elements as well as in style property manipulation

Cross-browser

Chrome, and more

Download jQuery
v1.9.0 32kB Minified

Download Un-minified Copy
View Source on GitHub →

How jQuery Works →

jQuery

http://jquery.com/

What is jQuery?

jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensbility, jQuery has changed the way that millions of people write JavaScript.

Who's Using jQuery

Resources

- jQuery Blog
- Contribute to jQuery
- About the jQuery Foundation
- Browse or Submit jQuery Bugs

Other jQuery Foundation Projects

メリットと役割のおさらい

①DOM APIを気にせず簡単に記述できる

write less, do more.

DOM API の煩雑さを避けて使える

```
elemNode.parentNode.removeChild(elemNode);
```

↓

```
$(elemNode).remove();
```

②クロスブラウザ対応の複雑さを回避できる

IE, Firefox, Safari, Opera, Chrome…

昔と今のAPIの違いや、特異な振る舞いの吸収

Msxml.XMLHTTP? attachEvent?

↓

$.ajax, $el.bind/$el.on

③大きいコミュニティと
プラグインの充実

Useful and Awesome Plugins!

プラグインがあれば何でもできる

User × Community × Ecosystem

↓

```
$.fn.awesomePlguin(‘feel good!’);
```

かつての問題は
jQueryが解決した

事実上の標準ライブラリ

次に取り組むべき昨今の問題

フロントエンド実装の現状と変化

Web サイト

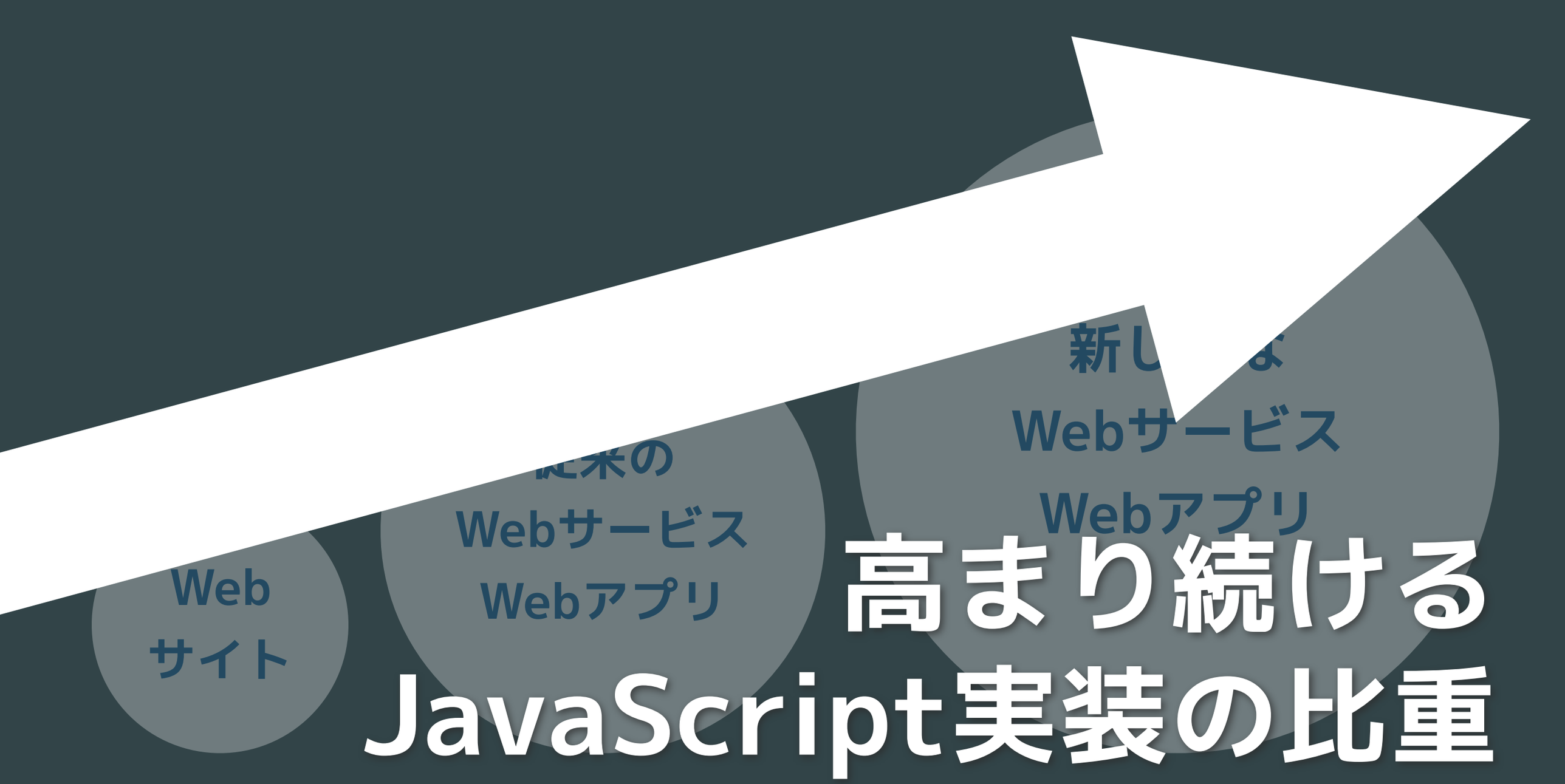
静的HTML
CMS利用

従来の Webサービス Webアプリ

リッチな
インターフェース

新しめな Webサービス Webアプリ

シングルページ
フルAjaxでシームレス



高まり続ける JavaScript実装の比重

静的HTML
CMS利用

リッチな
インターフェース

シングルページ
フルAjaxでシームレス

大きくて複雑で広域な JavaScriptの増加

Code Bloat & Logic Complexity

スマフォ向けサービスの例

フルAjaxシングルページ

JSファイル **100** 超　総行数 **20,000** 超

jQueryで解決できない問題

どんどん増えるJavaScriptの行数
以前に書かれたコードの意味がよくわからない
メンテナンスの度に内容の解析から始まる
付け足して汚くなっていくコード　　etc...

これらを解決するためには？

アーキテクチャ設計が重要

Architecture Design

アーキテクチャ設計？

ひとつの大きな塊ではなく

構造があって整理されていること

設計があいまいだと...?

ひとつの大きいJavaScriptが出来上がる

ある関数がどこに依存しているか分からない

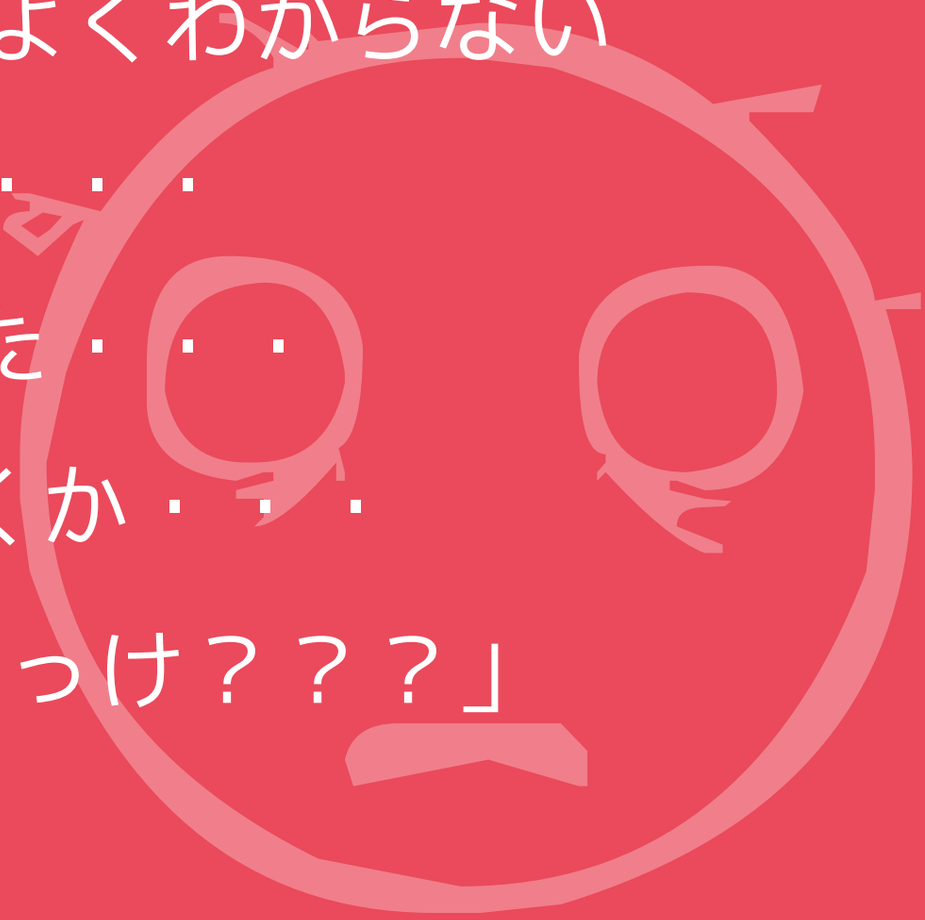
そもそも $(‘selector’)の羅列でよくわからない

コメントがあっても探しにくい・・・

こっちを直したらあっちが壊れた・・・

このページだけscript要素に書くか・・・

（1ヶ月後）「このscriptなんだっけ？？？」



無理なスケジュール
失われたモチベーション

理性や正気を失う原因

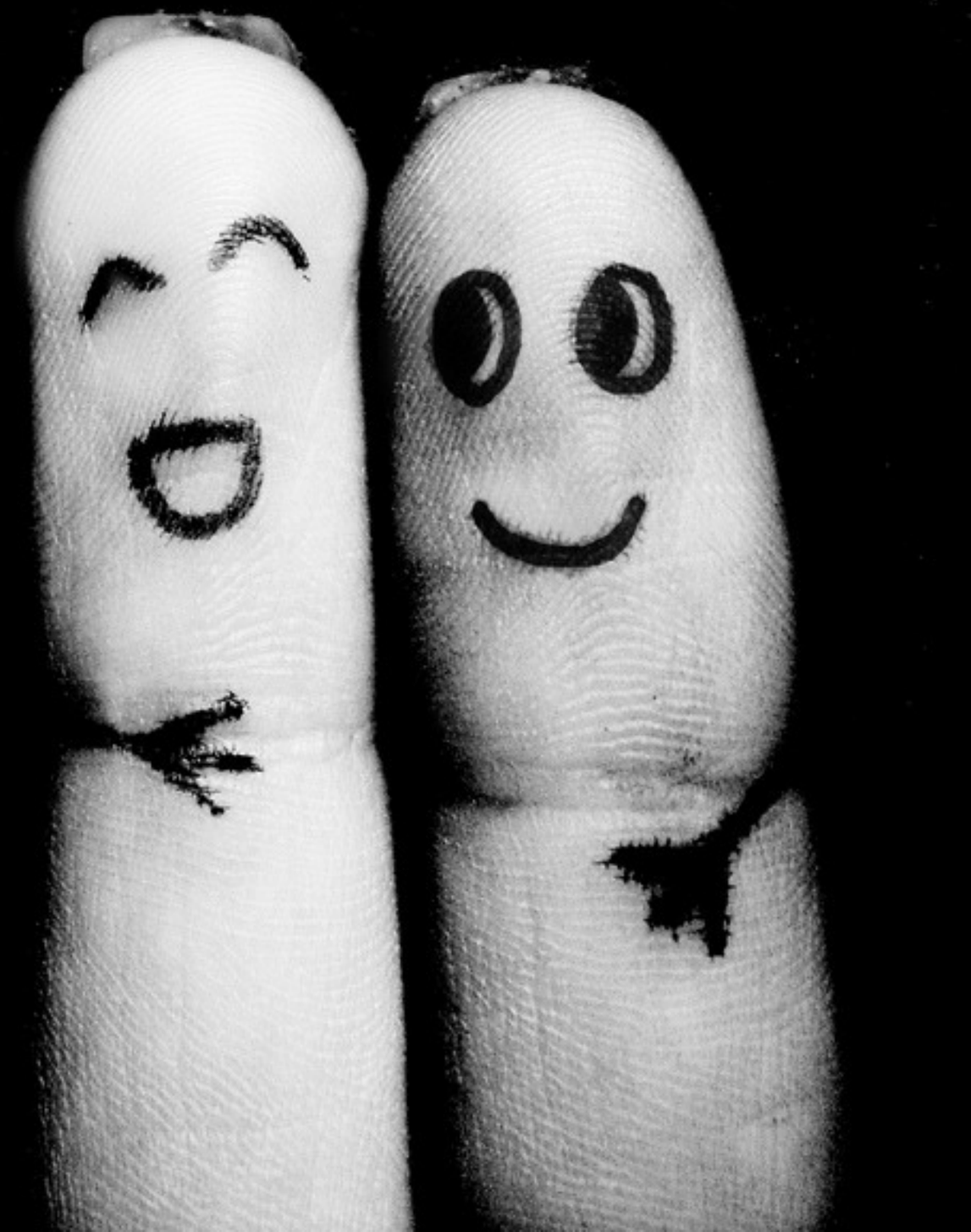
つらくなるjQuery（よく見る）

$(selector).abc(.... が延々と続く技術的負債

```
$(document).ready(function() {
  $('article.left section').click(function() {
    var was_selected = $(this).hasClass('section-selected');
    $('article.left section').removeClass('section-selected');
    if (!was_selected) {
      $(this).addClass('section-selected');
    }
});

$('article.right section').click(function() {
  $(this).toggleClass('right-selected');
  if ($('section.right-selected')) {
    $(this).children('input.choose').toggle();
  }
});

$('input.choose').click(function() {
  var section = $('section.section-selected');
```

いわゆる
ベタ書き

読みたくない



Juliana Coutinho

設計がしっかりしてると...!

JavaScriptが機能単位で分割されている

あれとこれの依存関係が明示されている

処理に名前がちゃんと付いていて見通せる

どこで何をやっているか探し出しやすい

他人のコードでも分かりやすい

自分のコードを他人に任せやすい

分担して早く帰れるように！

どうやって取り組む？

jQueryに頼れない・・・

jQuery to Backbone

Backbone.jsと
アーキテクチャとMVC

Backbone.js (0.9.10)

» GitHub Repository
» Annotated Source

Introduction

Upgrading

Events

- on
- off
- trigger
- once
- listenTo
- stopListening
- 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
- validate
- validationError
- url
- urlRoot
- parse

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 tickets to Lighthouse.

Backbone is an open-source component of DocumentCloud.

Backbone.js

http://backbonejs.org/

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

Development Version (0.9.10)	*56kb, Full source, lots of comments*
Production Version (0.9.10)	*6.3kb, Packed and gzipped*
Edge Version (master)	*Unreleased, use at your own risk*

build status passing

Initial Release
2010/10/13
Gzipped Size
6.3KB
Latest Version
1.0.0

構造化をサポートするライブラリ

View, Model, Router等を備える

海外の有名サイトも
使うくらいにメジャー

Hulu, Foursquare, Walmart, Linkedin Mobile…

今年に入って
ついに ver 1.0.0 へ

β版シンドロームから卒業

依存するライブラリ

Selector Based Library

- ✓ **jQuery**
- ✓ Zepto.js (lightweight clone)

Utility Belt Library

- ✓ **Underscore.js**
- ✓ Lodash (more faster)

Todo MVC

http://addyosmani.github.com/todomvc/

Introduction

Developers these days are spoiled with choice when it comes to selecting an **MV* framework** for structuring their JavaScript web apps.

Backbone, Ember, AngularJS, Spine... the list of new and stable solutions continues to grow, but just how do you decide on which to use in a sea of so many options?

To help solve this problem, we created **TodoMVC** - a project which offers the same Todo application implemented using MV* concepts in most of the popular JavaScript MV* frameworks of today.

Tweet 767 +1

JavaScript

Backbone.js	Knockback.js	Sammy.js	PlastronJS
Ember.js	Maria	Epitome	rAppid.js
KnockoutJS	cujo.js	soma.js	Funnyface.js
Dojo	dermis	DUEL	Knockout + ClassBinding
YUI	Montage	PureMVC	
Agility.js	Ext.js	Olives	AngularJS (optimized)

Compile To JavaScript

Spine	Batman.js	TypeScript + Backbone.js	Closure

Home | Download / Install | Tutorials | Live examples | Documentation | Forum | Sou

Knockout.

Simplify dynamic JavaScript UIs by applying the Model-View-View M (MVVM) pattern

Download
v2.2.1 - 14kb min+gz

y concepts

Knockout

http://knockoutjs.com/

Declarative Bindings
Easily associate DOM elements with model data using a concise, readable syntax

Automatic UI Refresh
When your data model's state changes, your UI updates automatically

Dependency Tracking
Implicitly set up chains of relationships between model data, to transform and combine it

Templating
Quickly generate sophisticated, nested UIs as a function of your model data

ore features

- Free, open source (MIT license)
- Pure JavaScript — works with any web framework

New: Interactive tutorials

Get started with knockout.js quickly, learning to build *single-page applicatio* *custom bindings* and more with these interactive tutorials.

batman.js

Overview Download Examples Documentation Commun

uild web apps beautifully and quickly

tman.js is a framework for building rich web applications with CoffeeScript
JavaScript. App code is concise and declarative, thanks to a powerful
stem of view bindings and observable properties. The API is designed with
veloper and designer happiness as its first priority.

ant to learn more? Check out **all the features** or some **examples**.

Download batman.js

Latest version: 0.11.2
Updated: July 17th, 2012
Patch release addressing some recent

View the release notes or the code.

batman.js

http://batmanjs.org/

ere's some code:

```
lass Shopify extends Batman.App
 @root 'products#index'
 @resources 'products'

lass Shopify.Product extends Batman.Model
 @persist Batman.RestStorage
```

From the Blog

Batman.js 0.9.0: The biggest thing since
Killer Croc

Posted 02 April 2012 by Nick Small

batman.js 0.9.0 is available on github and n

ember

ABOUT GUIDES API COMMUNITY BLOG FORK US

A framework for creating
ambitious web applications.

DOWNLOAD EMBER 1.0.0-PRE.4

47k min+gzip | minified

Ember.js

http://emberjs.com/

MORE PRODUCTIVE OUT OF THE BOX.

Write dramatically less code with

Don't waste time making trivial

Ember.js is built for productivity

GULARJS Home Learn Develop Discuss Search

HTML enhanced for web apps!

View on GitHub

Download (1.0.4/1.1.2)

Follow +AngularJS on +1 +14116

Follow @angularjs 8,732 followers

Follow 8,732 fo

ny AngularJS?

Alternatives

Extensibility

AngularJS

http://angularjs.org/

Repositories Members

Find a Repository…

All Sources Forks Mirrors

aura

A scalable, event-driven JavaScript architecture for developing widget-based applications. Works wi
Backbone.js and other frameworks.

Last updated 4 days ago

JavaScript ★ 1,921

aura-identity

Visual Resources for the Aura Project

Last updated 23 days ago

urajs

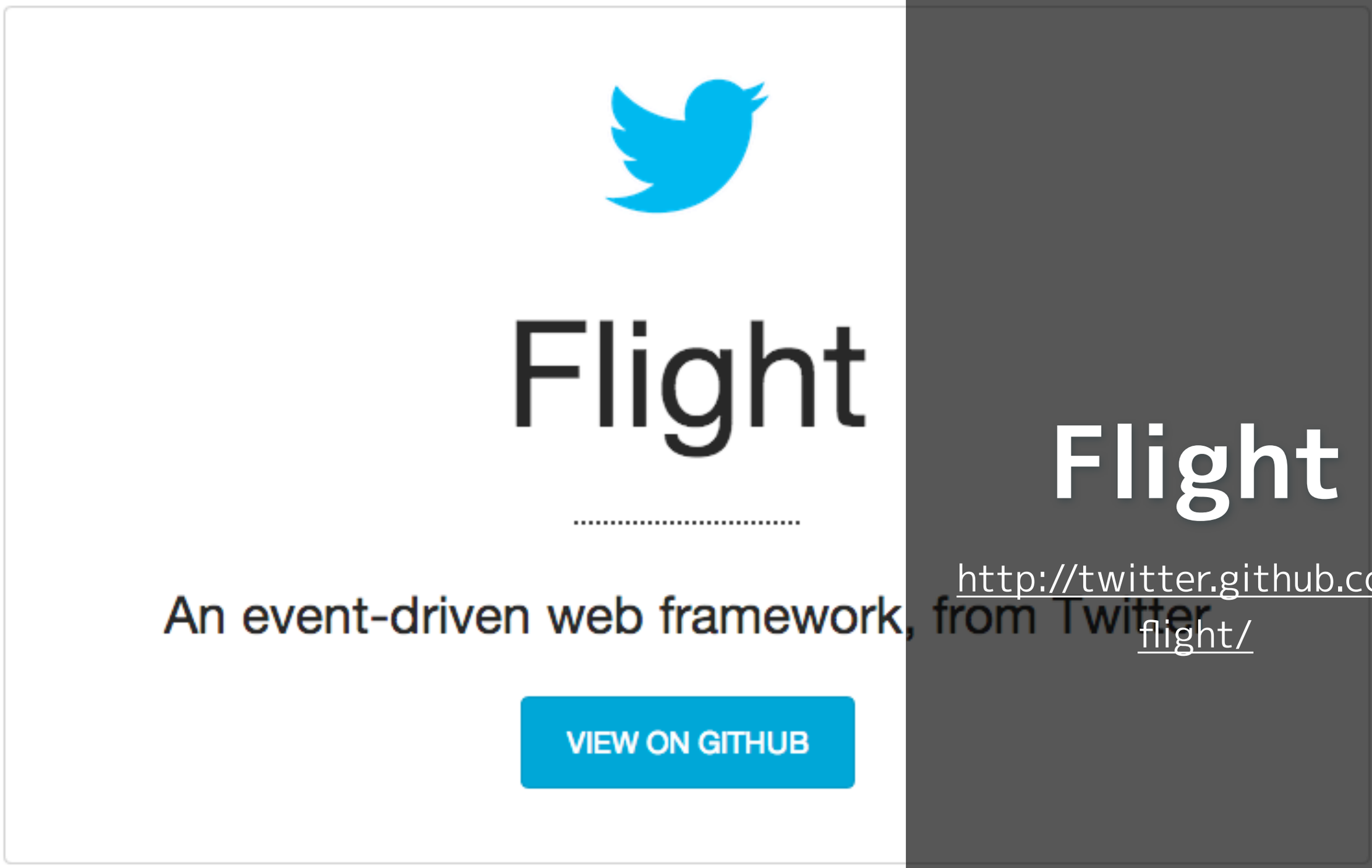
Joined on Dec 28, 2012

2 public repos
11 members

aura

★ 1

https://github.com/aurajs/aura



Flight

http://twitter.github.com/flight/

Backbone.jsは MVCフレームワーク？

MVCについての学習が必要か

用語の整理

アーキテクチャとMVCとBackbone.js

実装

Backbone.js
Ruby on Rails など

理論

オブジェクト志向(技法)
MVC(設計デザイン) など

問題

アーキテクチャ・設計の必要性

実装

Backbone.js
Ruby on Rails など

理論

オブジェクト志向(技法)
MVC(設計デザイン) など

問題

アーキテクチャ・設計の必要性

実装

Backbone.js
Ruby on Rails など

理論

オブジェクト志向(技法)
MVC(設計デザイン) など

問題

アーキテクチャ・設計の必要性

実装

Backbone.js
Ruby on Rails など

理論

オブジェクト志向(技法)
MVC(設計デザイン) など

問題

アーキテクチャ・設計の必要性

実装

Backbone.js
Ruby on Rails など

理論

オブジェクト志向(技法)
MVC(設計デザイン) など

問題

アーキテクチャ・設計の必要性

今回は、MVC云々の話は
意識せずお聞きください

BACKBONE.JS は、

✕

多機能さがウリの
堅牢なMVCフレームワーク

◎

アーキテクチャづくりを
サポートする小さいライブラリ

です。

it serves as a foundation for your application, you're meant to **extend and enhance it in the ways you see fit**

via. Backbone.js FAQ
http://backbonejs.org/#FAQ-why-backbone

MVCのことは気にしすぎないで
まずは使ってみることで
JSアーキテクチャを実践してみる

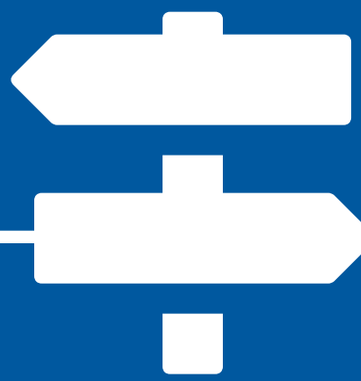
jQuery to Backbone

コードを構造化する

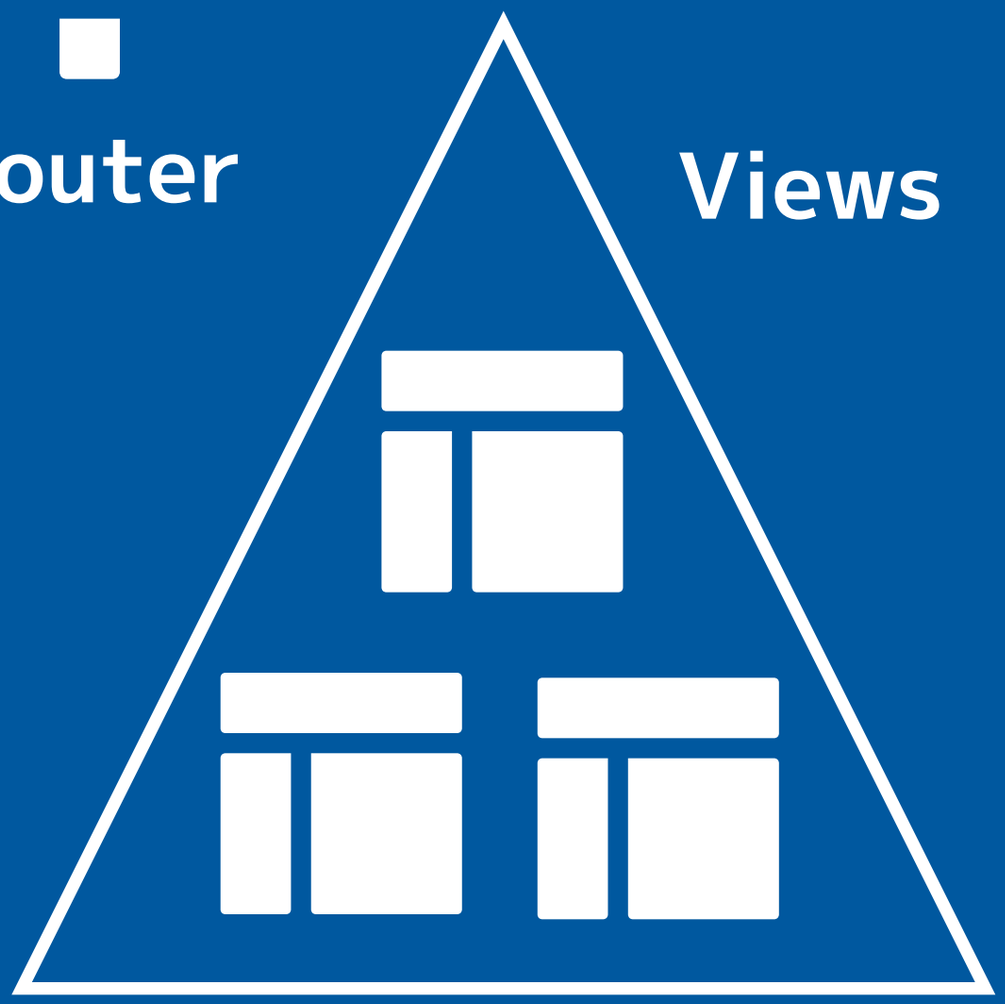
Backbone.jsにおける コンポーネント

View, Model, Collection, Router

Backbone



Router



Views

via. Backbonification - Migrating NewsBlur From DOM Spaghetti to Backbone.js
https://speakerdeck.com/samuelclay/backbonification-migrating-newsblur-from-dom-spaghetti-to-backbone-dot-js?slide=12

View

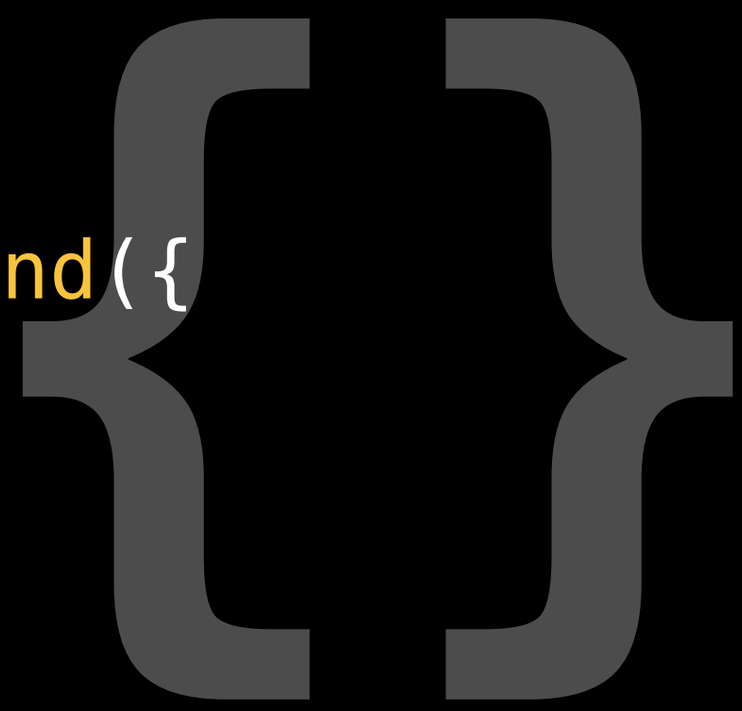
Backbone.View

見た目とUIにおける入出力

DOM要素の管理

ユーザー操作(イベント)制御

典型的なView



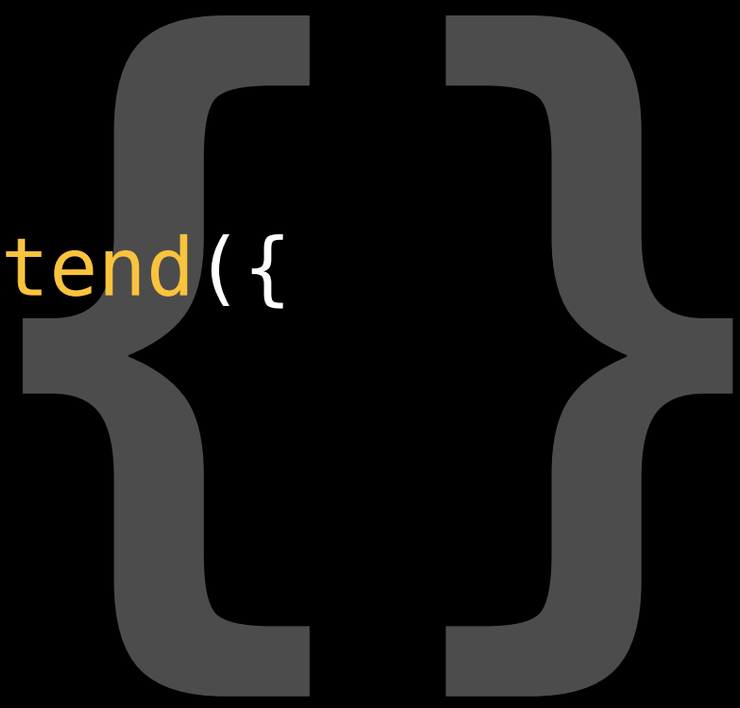
```
var AcmeView = Backbone.View.extend({
  events: {
    ‘click p’: ‘onClickButton’
  }
  onClickButton: function() {
    // triggered click event!
  },
  render: function() {
    this.$el.html(‘<p>Hello World!</p>’);
  }
});
var view = new AcmeView({el: ‘#main’});
view.render();
```

Model

Backbone.Model

取り扱うデータの一単位
ストレージとの通信・同期
APIや情報のレコードを表現

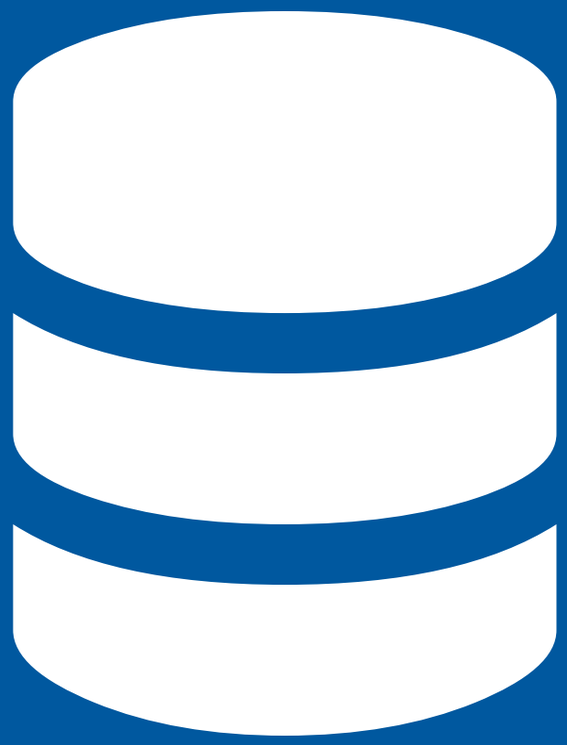
典型的なModel



```
var AcmeModel = Backbone.Model.extend({
  defaults: {},
  url: ‘api/v1/path/to’
});

var model = new AcmeModel();
```

```
var view = new AcmeView({model: model});
model.fetch({
  success: view.render
});
```

Collection



Backbone.Collection

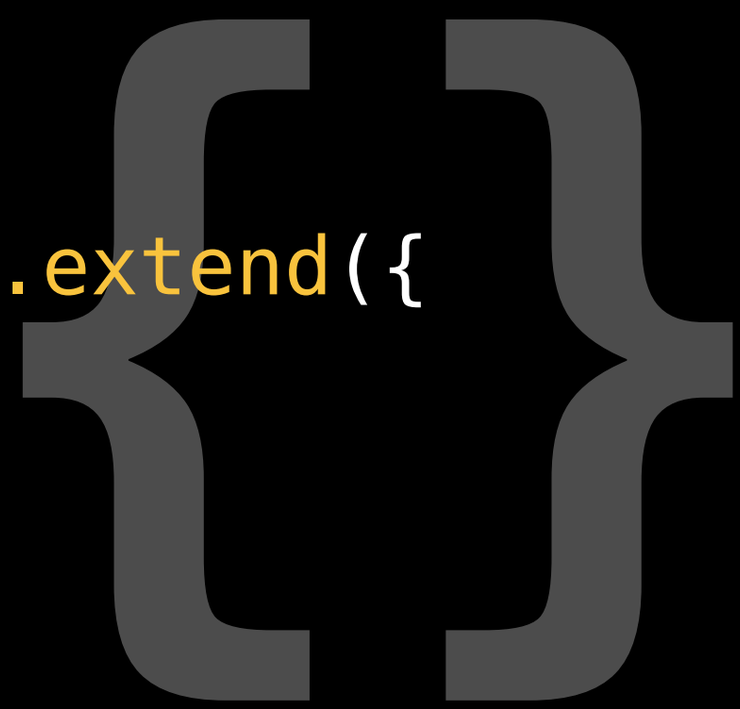
Modelが集合したリスト

リスト操作...where, filterなど

Modelと同様の通信・同期

※MVCのCはContollerなので無関係

典型的なCollection



```
var Persons = Backbone.Collection.extend({
  url: ‘api/v1/path/to’,
  model: Person
});
var persons = new Persons();
persons.fetch({
  success: function() {
    this.findWhere({
      name: ‘anonymous’
    }).sayName(); // ‘anonymous!’
  }
});
```

Router

Backbone.Router

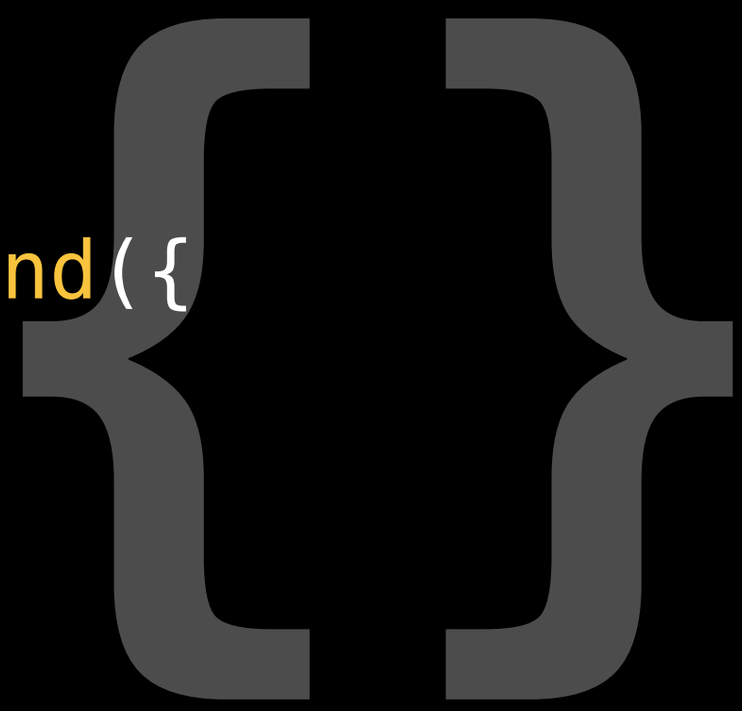
URLによる処理の振り分け

hashchange, pushstate

遷移処理のnavigate

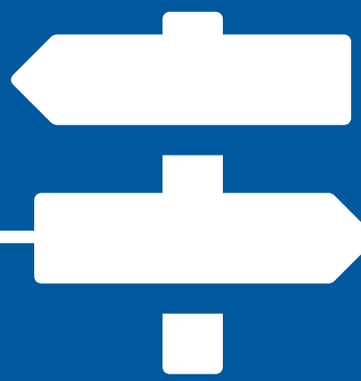
※しいていえばContollerっぽい役割なのがこれ

典型的なRouter

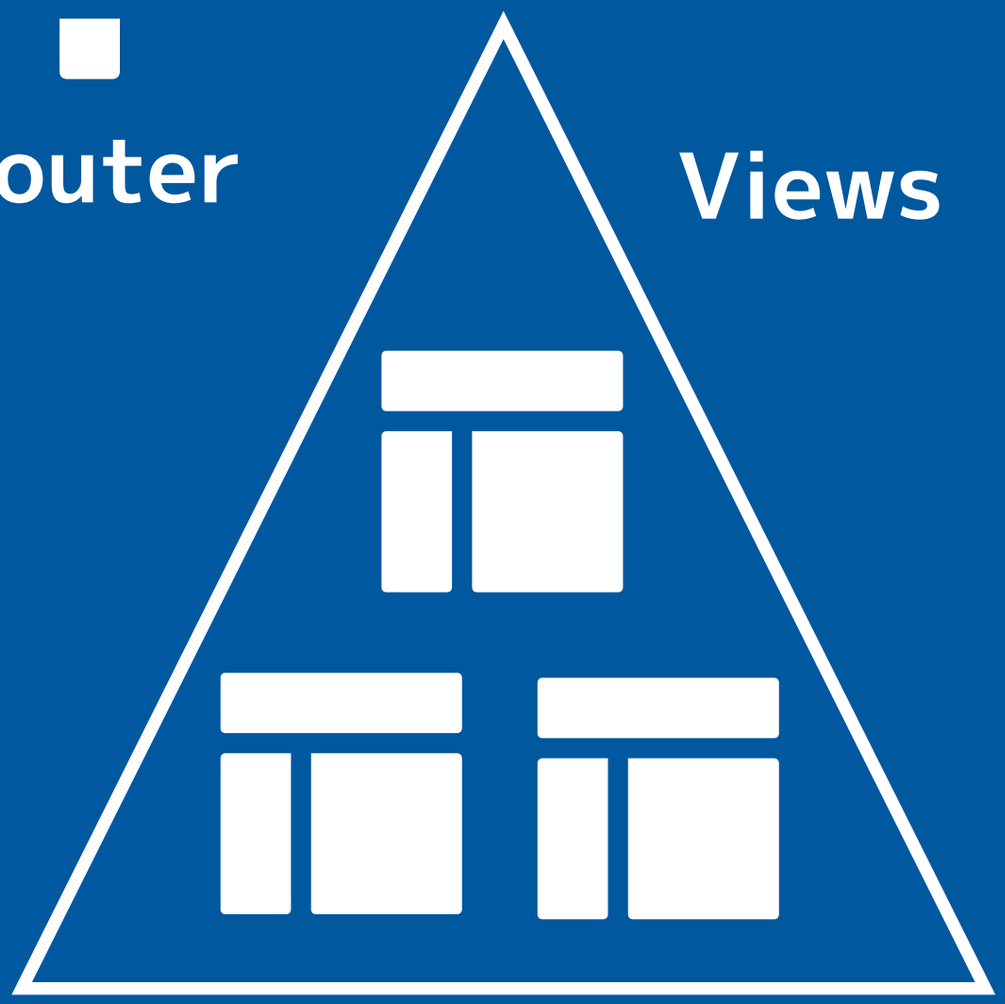


```
var Router = Backbone.Router.extend({
  routes: {
    'store/:storeId': 'gotoStore'
  },
  gotoStore: function(storeId) {
    new StoreView({
      model: new Store(storeId);
    });
  }
});
var app = new Router();
Backbone.history.start();
```

Backbone



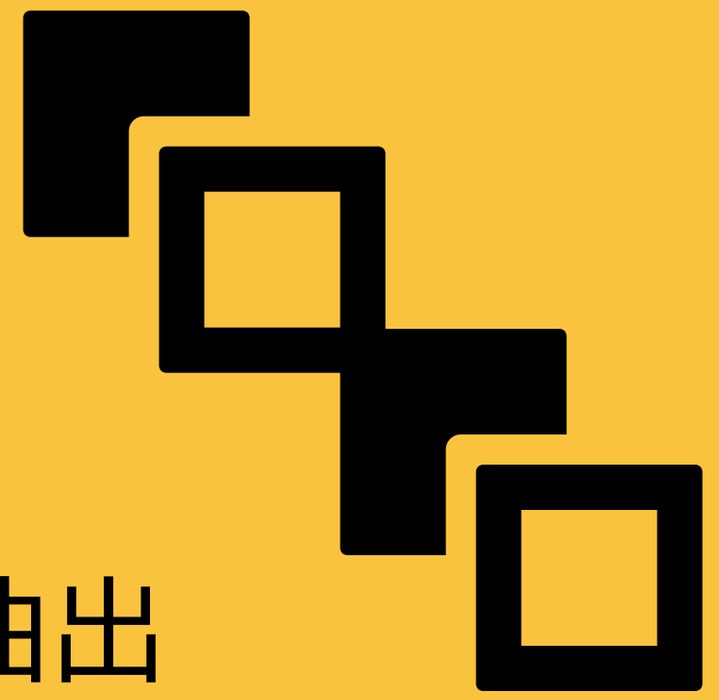
Router



Views

via. Backbonification - Migrating NewsBlur From DOM Spaghetti to Backbone.js
https://speakerdeck.com/samuelclay/backbonification-migrating-newsblur-from-dom-spaghetti-to-backbone-dot-js?slide=12

Backbone.jsを実際に使ってみる



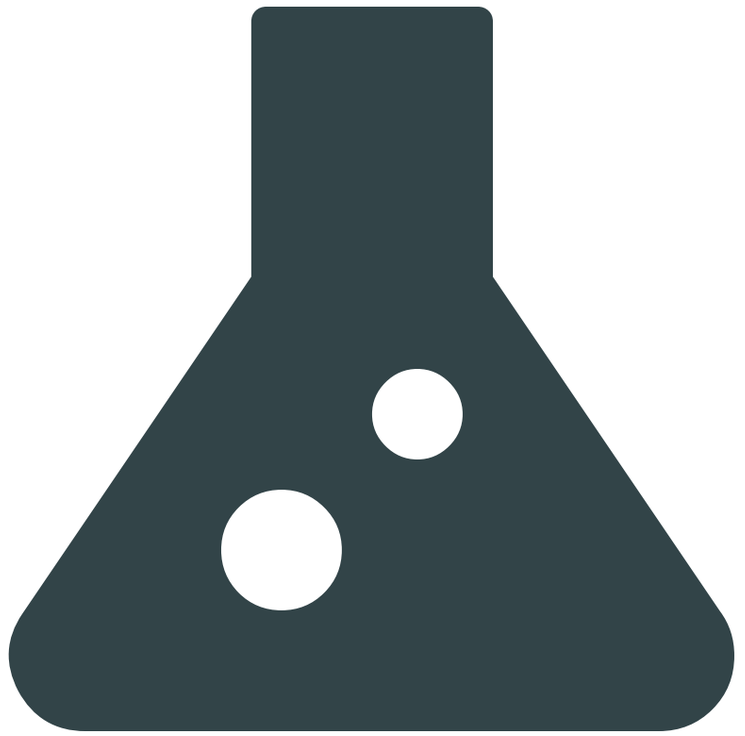
Viewの分離とメソッドの抽出

GitHub APIを使った Gistビューワー

実用性はさておき

DEMO

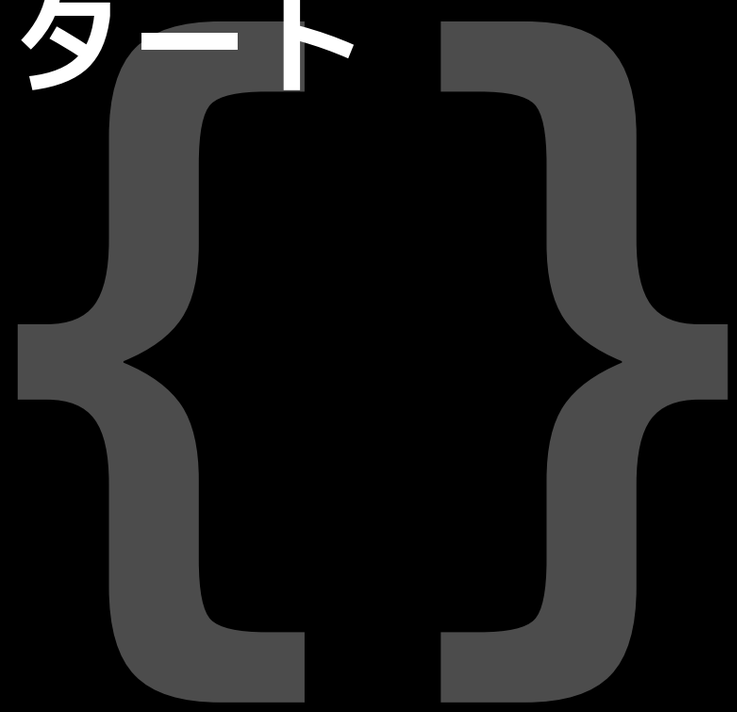
1. Backbone.Viewを作成
2. renderメソッドを抽出
3. テンプレートの分離
4. イベントの定義



実物＆初期コード紹介

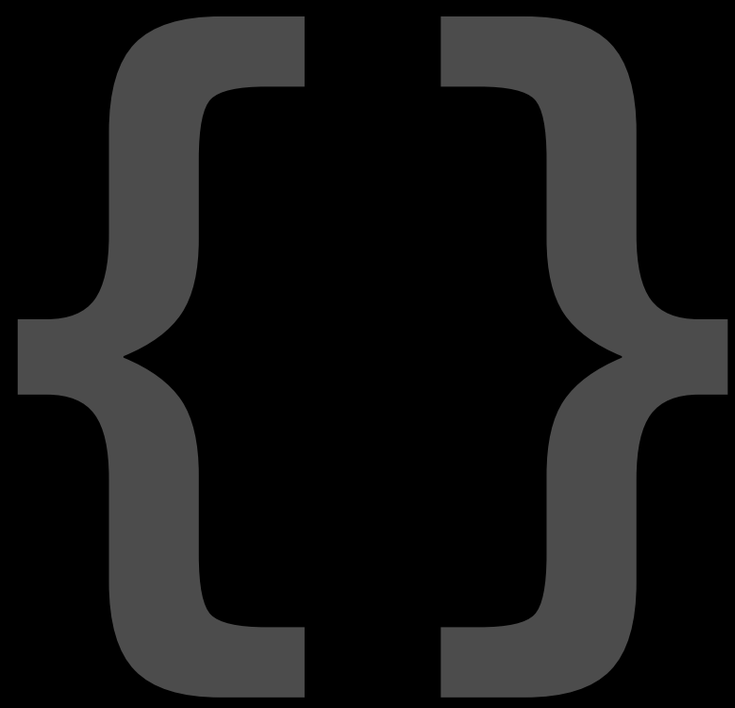
http://localhost:8000/

ピュアなjQueryコードからスタート



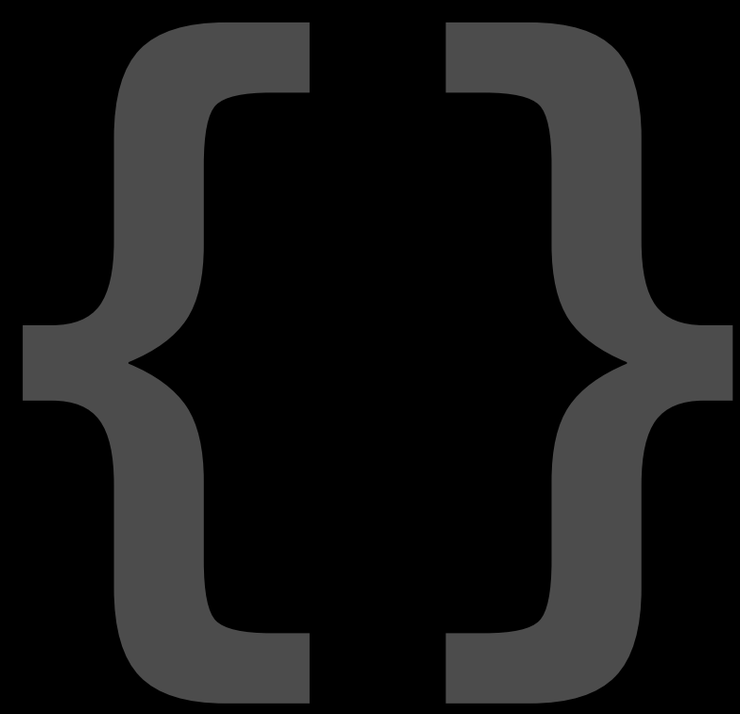
```
var $list    = $('#js-gists');
$.ajax({
  method: 'GET',
  url: 'https://api.github.com/gists',
  data: oauthData,
  dataType: 'json'
}).done(function(data) {
  var i = 0,
      html = '',
      item;
  while (item = data[i++]) {
    html += '<li>'+
      '<a data-src="'+item.url+'" href="#">'+item.description+'</a>'+
      '<a href="'+item.html_url+'">Show in gists</a>'+
      '</li>';
  }
  $list.html(html);
});
$list.on('click', '[data-src]', previewGist);
```

おもむろにViewを作成



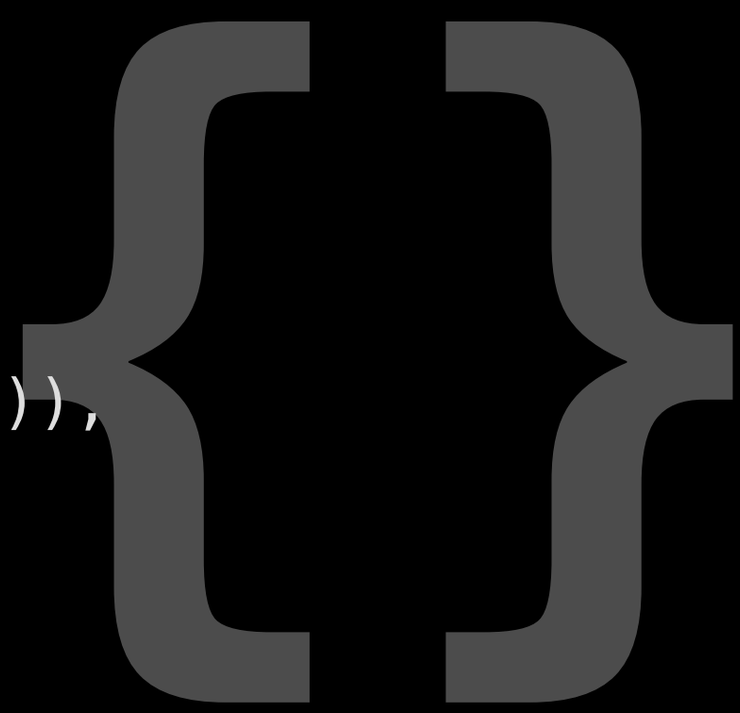
```
var GistsListView = Backbone.View.extend({
  el: '#js-gists',
  initialize: function() {
    var $list = this.$el;
    $.ajax({
      method: 'GET',
      url: 'https://api.github.com/gists',
      data: oauthData,
      dataType: 'json'
    }).done(function(data) {
      var i = 0, html = '', item;
      while (item = data[i++]) {
        html += '<li>'+
          '<a data-src="'+item.url+'" href="#">'+item.description+'</a>'+
          '<a href="'+item.html_url+'">Show in gists</a>'+
          '</li>';
      }
      $list.html(html);
    });
    $list.on('click', '[data-src]', previewGist);
  }
});
var gistsList = new GistsListView();
```

renderメソッドを抽出



```
var GistsListView = Backbone.View.extend({
  el: '#js-gists',
  initialize: function() {
    _.bindAll(this);
    $.ajax({
      method: 'GET',
      url: 'https://api.github.com/gists',
      data: oauthData,
      dataType: 'json'
    }).done(this.render);
    this.$el.on('click', '[data-src]', previewGist);
  },
  render: function(data) {
    var i = 0, html = '', item;
    while (item = data[i++]) {
      html += '<li>'+
        '<a data-src="'+item.url+'" href="#">'+item.description+'</a>'+
        '<a href="'+item.html_url+'">Show in gists</a>'+
        '</li>';
    }
    this.$el.html(html);
    return this;
  }
}); var gistsList = new GistsListView();
```

テンプレートの分離

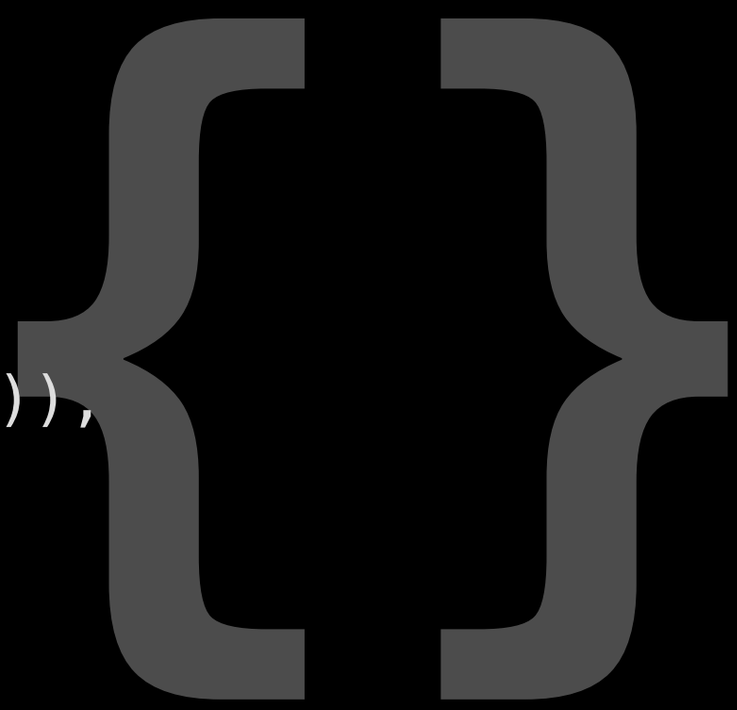


```
var GistsListView = Backbone.View.extend({
  el: '#js-gists',
  tmpl: _.template($('#tmpl-js-gists').html()),
  initialize: function() {
    _.bindAll(this);
    $.ajax({
      method: 'GET',
      url: 'https://api.github.com/gists',
      data: oauthData,
      dataType: 'json'
    }).done(this.render);
    this.$el.on('click', '[data-src]', previewGist);
  },
  render: function(data) {
    this.$el.html(this.tmpl({items: data}));
    return this;
  }
});
var gistsList = new GistsListView();
```

Underscoreテンプレート

```
<script id="tmpl-js-gists" type="tmpl/text">
<% _.each(items, function(item) { %>
<li>
  <a data-id="<%= item.id %>" data-src="<%= item.url %>">
    <%= item.description %>
  </a>
  <a href="<%= item.html_url %>">Show in gists</a>
</li>
<% }); %>
</script>
```

イベントの定義

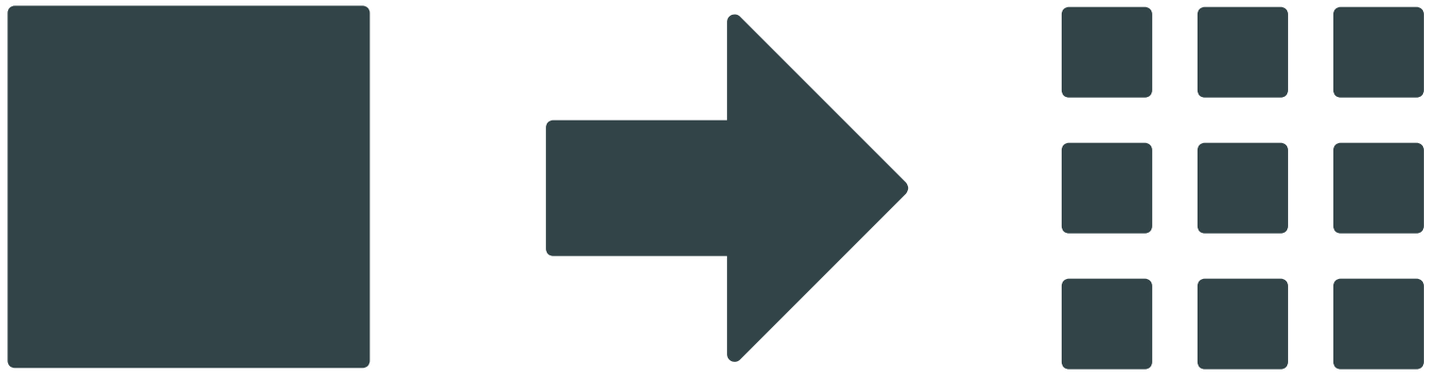


```
var GistsListView = Backbone.View.extend({
  el: '#js-gists',
  tmpl: _.template($('#tmpl-js-gists').html()),
  events: {
    'click [data-src]': previewGist
  },
  initialize: function() {
    _.bindAll(this);
    $.ajax({
      method: 'GET',
      url: 'https://api.github.com/gists',
      data: oauthData,
      dataType: 'json'
    }).done(this.render);
  },
  render: function(data) {
    this.$el.html(this.tmpl({items: data}));
    return this;
  }
});
var gistsList = new GistsListView();
```

大きな現実
小さな実装

現実の要件を、小さい実装に分割する

実装の分割によって



- ✓ コードの見通しが良くなる
- ✓ 個々のパーツの責任が明確になる
- ✓ 疎結合パーツは再利用できる

Backbone.jsの部品で分割すれば構造ができる

自然とアーキテクチャが生まれる

ModelとCollectionの利用

GitHub APIの関連処理を抽出

Gist

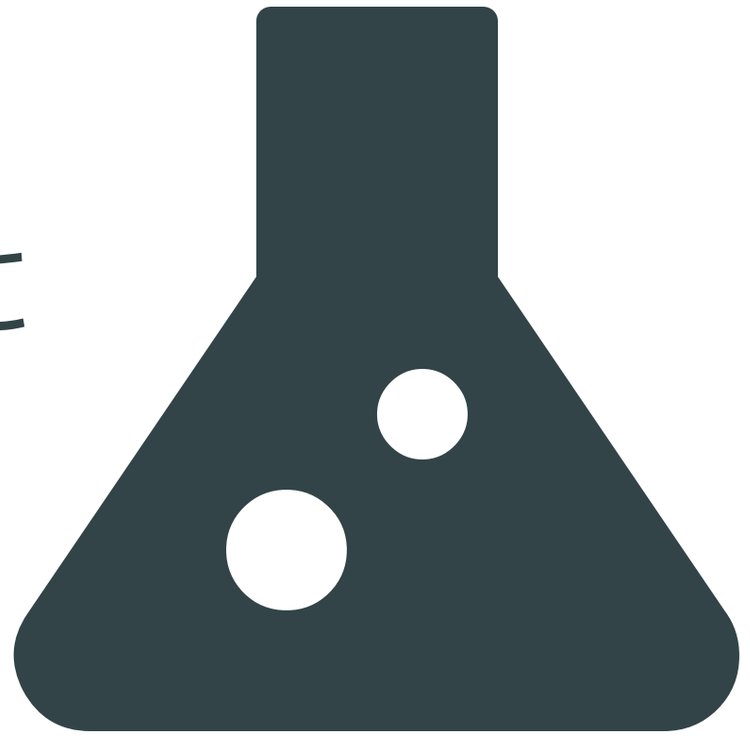
Gists

Model

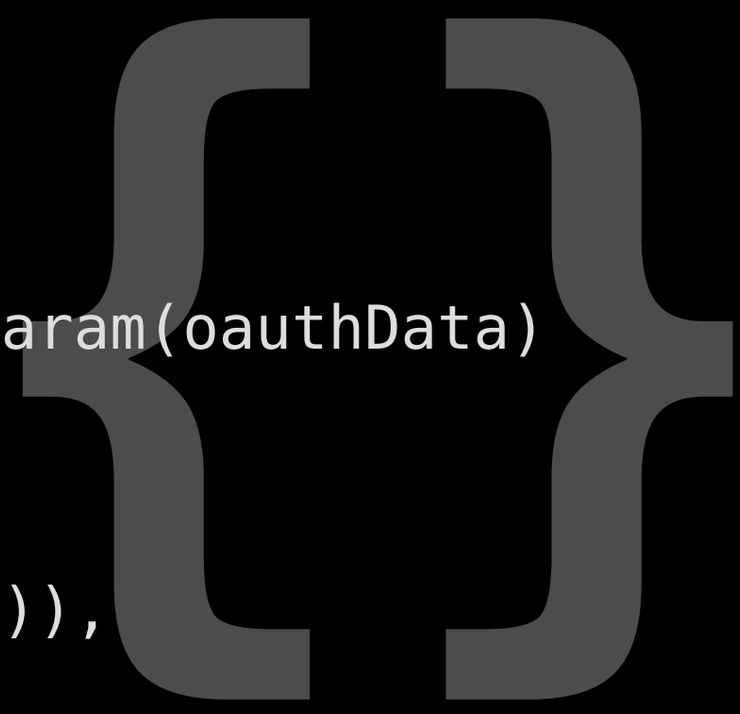
Collection

DEMO

今回はCollectionを中心に

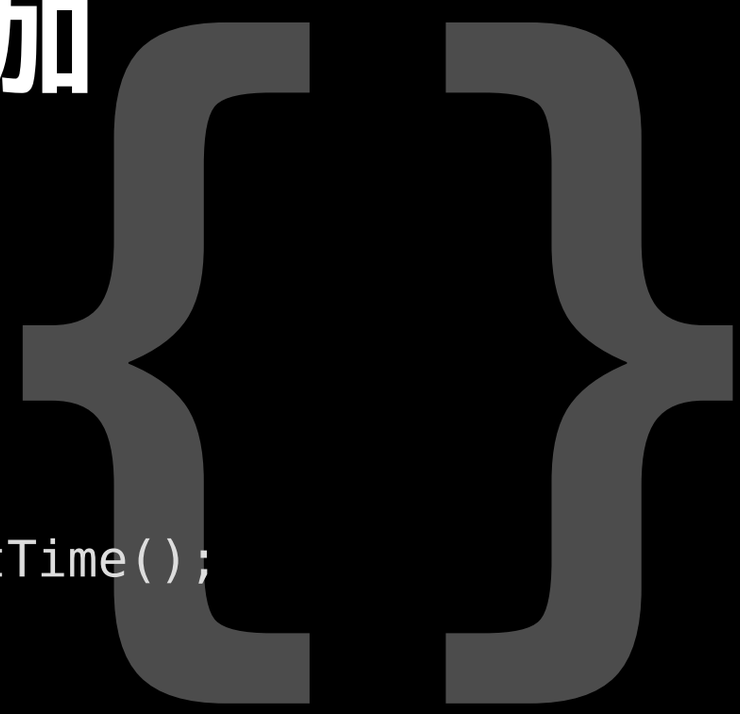


Collectionを作成



```
var Gists = Backbone.Collection.extend({
  url: 'https://api.github.com/gists?' + $.param(oauthData)
});
var GistsListView = Backbone.View.extend({
  el: '#js-gists',
  tmpl: _.template($('#tmpl-js-gists').html()),
  events: {
    'click [data-src]': 'preview'
  },
  initialize: function() {
    _.bindAll(this);
    this.collection.fetch({ success: this.render });
  },
  render: function() {
    this.$el.html(this.tmpl({items: this.collection.toJSON()}));
    return this;
  }
});
var gistsList = new GistsListView({ collection: new Gists() });
```

Collectionのソート機能を追加



```
var Gists = Backbone.Collection.extend({
  orderRule: 'updated_at',
  comparator: function(gist) {
    switch(this.orderRule) {
      case 'updated_at':
        return - new Date(gist.get('updated_at')).getTime();
    }
  },
  orderBy: function(rule) {
    this.orderRule = rule;
    this.sort();
  }
});
// partial of GistListView
events: {
  'click #js-sort-updated': 'sortByUpdatedAt',
},
initialize: function() {
  this.collection.fetch({success: this.render});
  this.collection.on('sort', this.render);
},
sortByUpdatedAt: function() {
  this.collection.orderBy('updated_at');
},
```

Backbone.jsを使うメリット

構造化はもちろん...

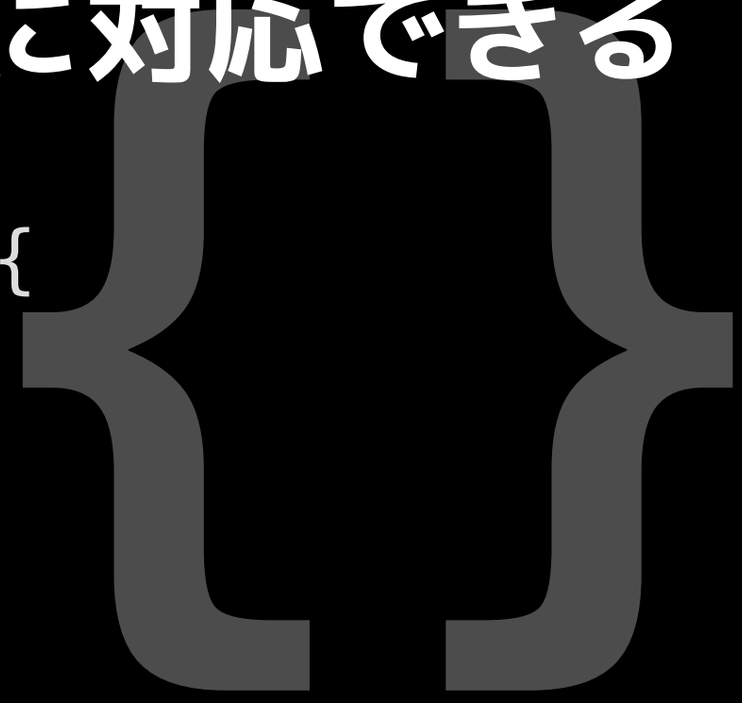
役割（ViewやModel）で確実に分割できる

```
var IconView = Backbone.View.extend({
  events: {
    "click .icon": "open"
  },
  render: function() {
    ...
  }
});
var Sidebar = Backbone.Model.extend({
  promptColor: function() {
    var cssColor = prompt("Please enter a CSS color:");
    this.set({color: cssColor});
  }
});
```

Backboneらしい規則性のあるコードになる

```
var DocumentRow = Backbone.View.extend({
  tagName: "li",
  className: "document-row",
  events: {
    "click .icon":          "open",
    "click .button.edit":   "openEditDialog",
    "click .button.delete": "destroy"
  },
  initialize: function() {
    this.listenTo(this.model, "change", this.render);
  }
  render: function() {
    ...
  }
});
```

制約がゆるい分、色々な構成に対応できる



```
var ListView = Phalanx.View.extend({
  components: {
    'moreBtn': ReadMoreBtnComponent
  },
  listeners: {
    'success moreBtn': 'renderMore'
  },
  ui {
    list: null
  }
  renderMore: function(html) {
    this.$ui.list.append(html);
  }
});
var listView = new ListView({el: '#js-list'});
```

ありがちな罠と
気をつけるべきこと

DRYとYAGNIとか...

DRYの罠

「Don’t Repeat Yourself」

繰り返しあらわれる処理について、
たとえば関数などにして、使い回せるようにする基本。

複雑さで多様性に対応しようとしてやりすぎると・・・

タイプに応じて何か計算してくれる関数

```
function awesomeCalc(type, count) {
  switch(type) {
    case ‘foo’: return count * 0.05;
    case ‘bar’: return count * 1.05;
  }
}
```

```
awesomeCalc('foo', 100); // => 5
```

タイプが増えたら引数も増えた

```
/**
 * @param {String} type
 * @param {Number} count
 * @param {Number} [fact] ←quxのときだけ必要
 */
function awesomeCalc(type, count, fact) {
  switch(type) {
    case ‘foo’: return count * 0.05;
    case ‘bar’: return count * 1.05;
    case ‘qux’: return count + 13 / fact;
  }
}
```

また増えた

```
function awesomeCalc(type, count, fact, n) {
  switch(type) {
    case ‘foo’: return count * 0.05;
    case ‘bar’: return count * 1.05;
    case ‘qux’: return count + 13 / fact;
    case ‘lol’: return count + 2 * n / fact;
  }
}
// 何の引数でしたっけ...(´ω`;)
awesomeCalc(‘foo’, 30, 3, 4);
```

別の関数をつくるか、工夫するかしよう！

```
function awesomeCalc(count, typeOrFunc) {
  switch(typeOrFunc) {
    case ‘foo’: return count * 0.05;
    case ‘bar’: return count * 1.05;
    default   : return count + typeOrFunc();
  }
}
// たとえば特異の係数は、呼び出し側に預けたりとか？

var fact = 5, n = 3;
awesomeCalc(10, function() {
   return 2 * n / fact;
});
```

YAGNIの原則

「You Aren't Going to Need It」

機能や実装は、本当に必要になるまで作らない。
慣れてきた頃にやってしまいがちな

いわゆる早すぎる最適化に近くて・・・

この処理はまた出てくる気がするぞ
汎用的に使えるようにしておこう！

実際の用途が出てきてもいないのに
『想像上の汎用性』に時間をかけてしまう

コードレベルだけでなく、
サービスやプロダクトを開発する際にも
そのときに良かれと思って付けた機能は、
必要とされることのほうが少ない。

無駄な機能がバグを呼び、
無駄な機能のために増えたコードは、
作業者にとって邪魔以外の何物でも無い。

ライブラリの導入とオーバーヘッド

パフォーマンスは？

jQueryにも言えるが
使い方によっては簡単に重くなる

```
// ループ内でappendすんな、毎回セレクタ走らせんな!!
$.each(persons, function(person) {
  $(‘ul’).append(‘<li>’+person.name+’</li>’);
});
```

ユーザーとして
ライブラリを効率良く使う

ライブラリの努力を無駄にしない！

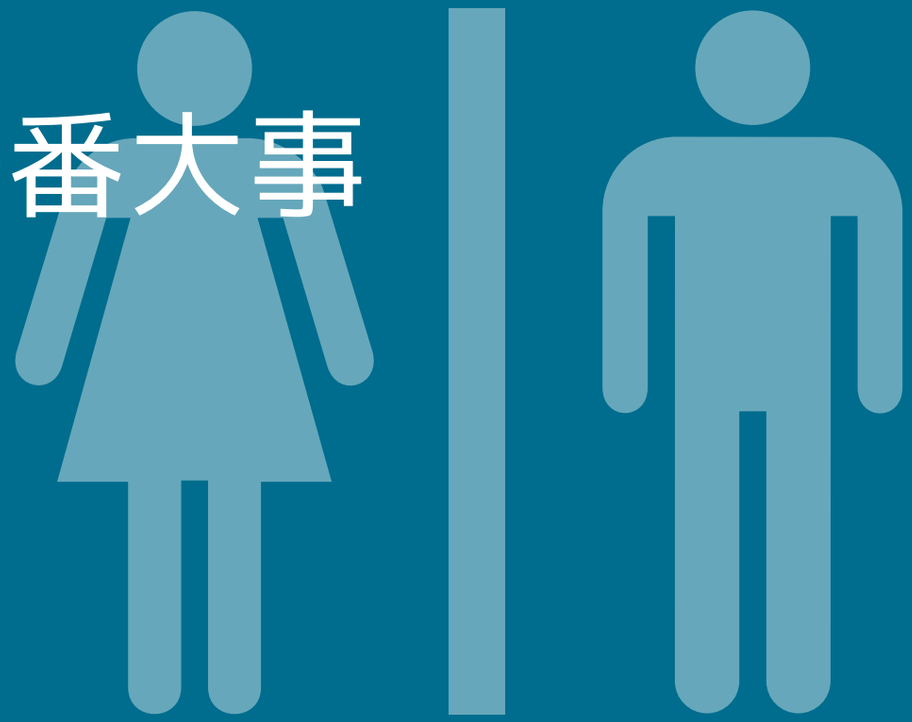
まとめ と おまけ

Backbone has made me a better programmer

via. Backbone has made me a better programmer | Float Left
http://floatleft.com/notebook/backbone-has-made-me-a-better-programmer

良い習慣のために
とりあえず分けてみる

「ものはためし」が一番大事



アーキテクチャの設計は
手を動かしてみるのが一番

自分で良い方法を選んで
組めるようになるのが大事

派生ライブラリ

There's More Than One Way To Do It

用途に合わせて拡張された具体例

Marionette

http://marionettejs.com/

Make your Backbone applications dance!

Marionette - 主にView拡張（1/2）

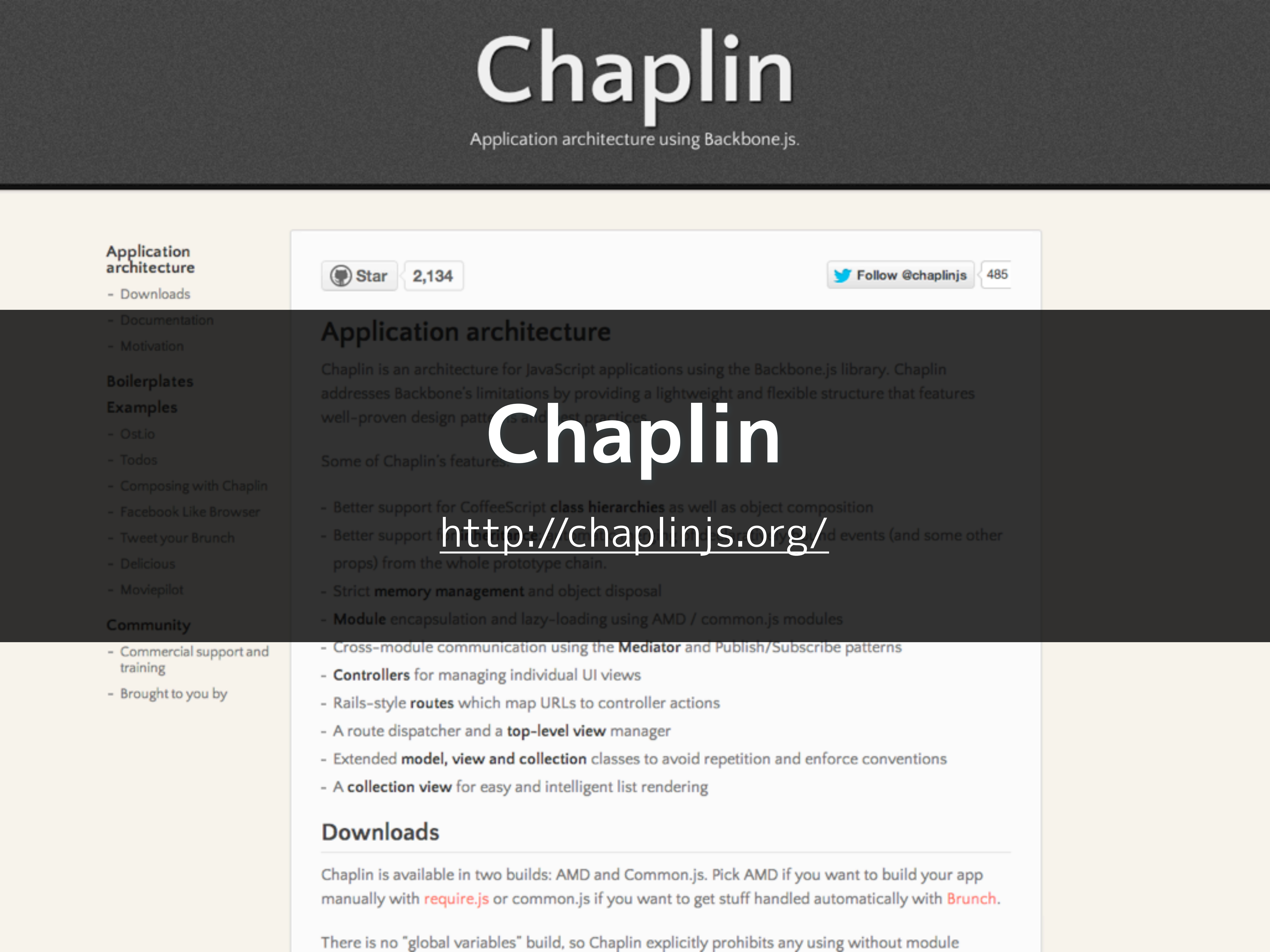
```
// Viewを大枠で管理する仕組み（以下はNested Layout）
MyApp = new Marionette.Application();
MyApp.addRegions({
  mainRegion: "#main"
});
var layout = new AppLayout();
MyApp.mainRegion.show(layout);
layout.show(new MenuView());

// よくあるViewの役割を上手に一般化して、再構築している
MyItemView = Marionette.ItemView.extend({});
Marionette.CollectionView.extend({
  itemView: MyItemView
});
```

Marionette - 主にView拡張（2/2）

```
// 頻出するバインディングを、より宣言的に見通しよく書ける
Marionette.View.extend({
    ui: {
        button: '.button'
    },
    events: {
        'click .button': 'onClickbutton'
    },
    modelEvents: {
        'change': 'onModelChange'
    },
    collectionEvents: {
        'add': 'onCollectionAdded'
    },
    triggers: :{
        'click .eventTrigger': 'do-something'
    }
```

Chaplin

http://chaplinjs.org/



Chaplin - 全体的な機能強化 (1/2)

```
# まさかのCoffee Script文化（余談：開発者はLiveScriptのひと）

# Chaplin.Viewで提供されるpassメソッドによる 1-way binding
pass: (selector, attribute) -&gt;
  return unless @model

  $el = @$ selector
  $el.text @model.get attribute if $el

  @listenTo @model, "change:#{attribute}", (model, value) =>
    @$(selector).text value

# 自身のViewで`@pass 'セレクタ', 'モデルの属性名'`とする
# Modelに変化があると、自動で要素のテキストも更新されるようになる
initialize: ->;
  @pass '.name', 'name'
  @pass '.phone', 'phone'
```

Chaplin - 全体的な機能強化 (2/2)

```
# RquireJSに依存していたり、RouterのあとにControllerがあったり
define [
  'controllers/controller',
  'models/likes',            # the collection
  'models/like',             # the model
  'views/likes-view',        # the collection view
  'views/full-like-view'     # the view
], (Controller, Likes, Like, LikesView, FullLikeView) -&gt;
'use strict'

class LikesController extends Controller
  beforeAction: (params, route) -&gt;
    if route.action is 'show'
      @redirectUnlessLoggedIn()
  index: (params) -&gt;
    @collection = new Likes()
    @view = new LikesView {@collection}
  show: (params) -&gt;
    @model = new Like id: params.id
    @view = new FullLikeView {@model}
```

THORAX

Getting Started API Reference

An opinionated, battle-tested Backbone + Handlebars framework to build large scale web applications.

Clone Seed Project

SCREENCAST GETTING STARTED API REFERENCE

Thorax

http://thoraxjs.org/

USED BY

Walmart ASDA METALAB

Hello World

Thorax is used to build some of the biggest Backbone applications in the world but it's easy to get started.

```
var view = new Thorax.View({
  greeting: "Hello",
  template: Handlebars.compile("{{greeting}} world!")
});
view.appendTo('body');
```

Features

Hello World

Easy Data Binding

Context Control

Collection Rendering

jQuery and Zepto Integration

Event Enhancements

Form Handling

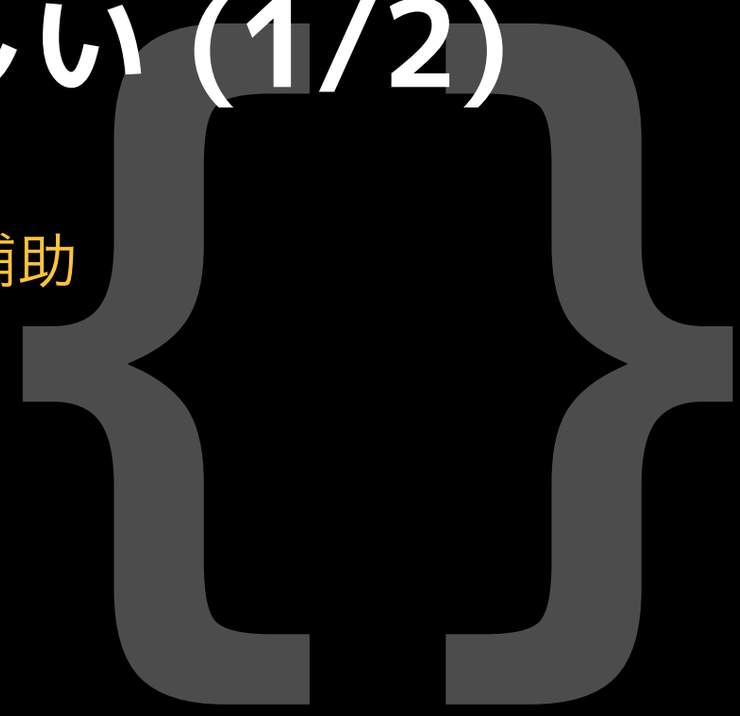
Quick Start

- Clone the Seed Project and start building your own application.
- Read about how Thorax works in the new Backbone Fundamentals Book.
- Install the Thorax Inspector Chrome extension.
- Building something smaller? Just link the core library as

Thorax - 特徴に乏しいが易しい (1/2)

```
// ウォルマートの内製部隊が開発しているBackbone + Handlebars の構成

// Eventsの拡張
var view = new Thorax.View({
  events: {
    rendered: function() {}
    model: {
      change: function() {}
    },
    ready: function() {},
    destroyed: function() {}
  },
  model: new Thorax.Model()
});
// Layoutの提供
var layout = new Thorax.LayoutView();
layout.appendTo('body');
layout.setView(view);
```

Thorax - 特徴に乏しいが易しい (1/2)



```
// Handlebarsのヘルパと連動したテンプレーティング補助
var view = new Thorax.View({
  collection: new Thorax.Collection([{
    title: 'Finish screencast',
    done: true
  }]),
  template: ...
});
```

```
{{#collection tag="ul"}}
  <li>
    <input type="checkbox" {{#done}}checked{{done}}
    {{title}}
  </li>
{{else}}
  <li>No todos yet.</li>
{{/collection}}
```

参考リソース

手を動かすときのお供に

ackbone / 194 commi

Merge pull request #41 from maepon/master ···

studiomohawk authored 6 days ago latest commit fddd40f3fb

docs	6 days ago	Merge pull request #41 from maepon/master [studiomohawk]
.gitmodules	2 months ago	change directory name [Layzie]
README.md	16 days ago	#39 Update README.md [ahomu]
backbone.js	2 months ago	Fix by supervisor's feedback [ahomu]
original @ c36df02	2 months ago	change directory name [Layzie]

en.ja OSS Backbone日本語訳

https://github.com/enja-oss/Backbone/

Backboneドキュメント日本語訳

部分的な翻訳済みドキュメントを集めていって、翻訳カバー率100%を目指します。

Pull Request・Issue・ご寄稿・ご指摘、いずれも歓迎しています。お気軽にどうぞ。このREADME文書自体の内容についても、ご指摘・ご提案があれば随時お知らせください。

en.ja-ossへの参加について

READMEレポジトリはenja-ossのトップページです。このレポジトリのREADMEやIssuesにて 本グループに参加いただく際の注意点などを記載しています。グループワークを円滑に行うために一読してください。

また参加いただく際にできればREADMEレポジトリをStarではなく、Watchいただければ。Google Groupなどを使った連絡についても

Developing Backbone.js Applications

By Addy Osmani @addyosmani

Star 5,562 Tweet 1,166

Was this helpful?

Developing Backbone.js Applications

http://addyosmani.github.com/backbone-fundamentals/

Questions?

http://aho.mu

@ahomu

github.com/ahomu

Photo Credits...thx♡

1. Two equestrian riders, girls on horseback, in low tide reflections on serene Morro Strand State Beach http://www.flickr.com/photos/mikebaird/2985066755
2. Energy Drinks - Monster, Red Bull and Rockstar http://www.flickr.com/photos/aukirk/8170825503
3. - Good Friends http://www.flickr.com/photos/ngmmemuda/4166182931
4. Rhino relaxation http://www.flickr.com/photos/macinate/2810203599
5. Whale backbone http://www.flickr.com/photos/vagawi/2257918524/
6. Sleeping 猫 http://www.flickr.com/photos/hansel5569/7687221498/
7. Alien vs Predator http://www.flickr.com/photos/steampirate/1056958115/