JavaScript Libraries You Aren't Using...Yet

Nathaniel T. Schutta
@ntschutta

First, a warning.

This is a survey!

10,000 foot view.

We won't get into much detail...

Goal is to whet your appetite.

What we'll cover...

What is it.

Look at some code.

Why you care.

Goal: discovery.

Monday - take one or two for a spin.

But we use jQuery.

You bet!

How many of you use jQuery today?

So do we.

Rocking good library.

Some would say it's a bit heavyweight.

Like, say, the jQuery core team.

https://groups.google.com/forum/#!topic/jquery-bugs-team/17rGK6eAAxI/discussion

http://blog.jquery.com/2011/11/08/building-a-slimmer-jquery/

1.7 deprecated a lot.

Are you using all of it?

Probably not.

Dragging along a lot of excess baggage.

Is that an issue?

It depends.

For desktop users?
Probably not.

But what about mobile?

Extra bits matter.

Especially on certain cell networks...

Movement today towards micro frameworks.

Do one or two things...

Really well.

Unix model!

Small, loosely coupled.

Pros:

If you're not using it, you don't need to push it.

Smaller learning curve.

Constraints shall set you free.

Cons:

May need multiple libraries.

Reinvent the wheel.

Existing skill sets.

Three rough categories.

jQuery alternatives.

JavaScript MVC.

JavaScript alternatives.

jQuip.

What is it.

jQuery in parts.

The good parts ;)

Re-org of jQuery.

Modular jQuery.

Provide the useful bits without the cruft.

jquip.js - 6.6 kb.

Automatically gets Sizzle if needed.

Minified and gzipped

- Events
- Document ready
- CSS
- Ajax
- Query engine (Sizzle, replacement)

What does it look like?

Well...jQuery.

Has a library builder.

Other things to consider...

Ender.

http://ender.no.de/

Package manager.

JSlim.

JavaScript optimizer.

https://github.com/zgrossbart/jslim

Built on Google Closure.

http://code.google.com/closure/compiler/

Uses Gradle.

MIT license.

Current version: 0.03.

Very early.

https://github.com/mythz/jquip

Zepto.js

"The aerogel-weight mobile JavaScript framework."

Thomas Fuchs.

Prototype, script.aculo.us and Ruby on Rails.

Minimalist library.

Goal: 5-10 kb.

~7.4 kb today.

About 10x smaller then jQuery and Prototype.

jQuery syntax.

Subset of features.

NOT a drop in jQuery replacement.

Mobile focus.

Works in “modern” browsers.

As in not IE.

Safari, Chrome, Firefox, Opera, any mobile WebKit.

```
$('p').bind('click', function(){
  $('span', this).css({color: 'red'});
});
```

Rich API.

- get
- first/last
- find
- parent/children/siblings
- next/prev
- filter
- show/hide
- html/text
- before/after/append/prepend
- css
- on/bind/live

And more.

Adds some utility functions.

Support for touch events.

Tap, double tap,
swipes, pinches.

Ajax support.

$.ajax(), $.get(),
$.post(), $.getJSON().

Includes $.os for environmental information.

Boolean for various OSes, string for version numbers.

Includes a Rake based build for minifying.

Uses UglifyJS.

MIT License.

Currently in beta.

Current version: 0.8.

http://zeptojs.com/

Backbone.js

Very lightweight.

As in ~5 KB compressed.

~1300 lines.

Fully documented.

Isn't a "UI" framework.

Built for MVC
JavaScript applications.

Models, events,
collections, views.

Controllers, persistence.

Influenced by Ruby on Rails.

Data lives in models.

Not the DOM.

Changes to models trigger change events.

Views are notified of said changes to the model.

Update accordingly.

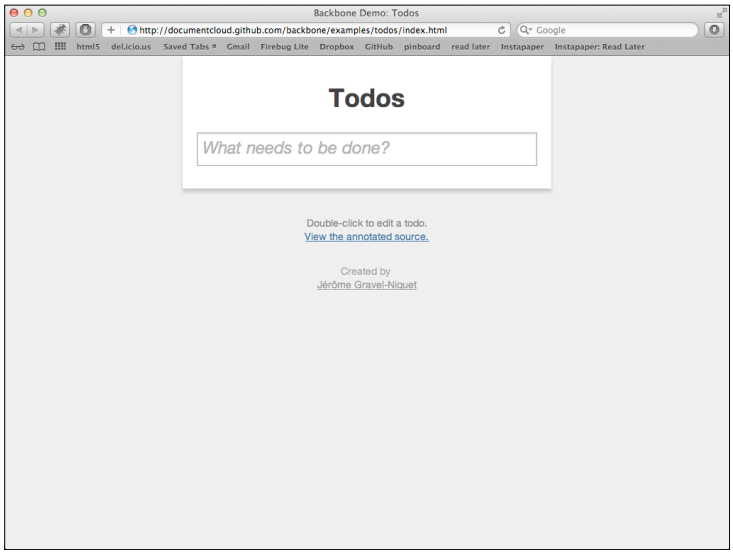
No more find stuff and change it - it just updates.

You'll be coding to events.

Several sample apps.

todos.js

http://documentcloud.github.com/backbone/docs/todos.html

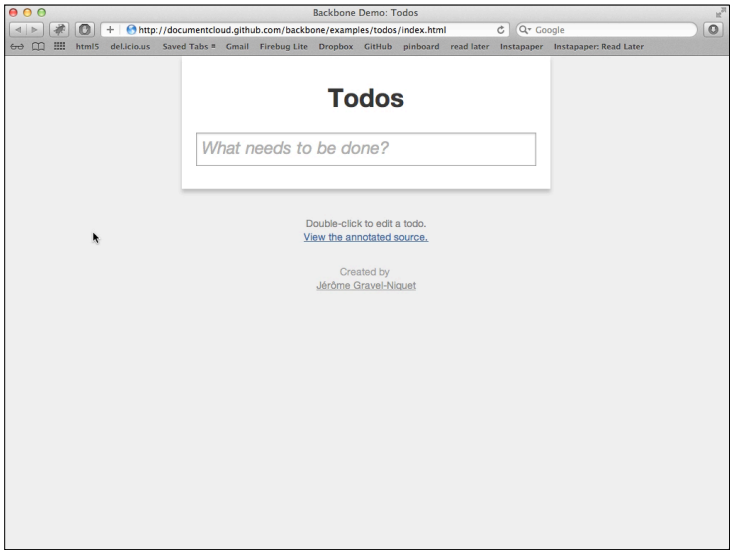



```
window.Todo = Backbone.Model.extend({

    defaults: function() {
      return {
        done:  false,
        order: Todos.nextOrder()
      };
    },

    toggle: function() {
      this.save({done: !this.get("done")});
    }

  });
```

```
window.TodoList = Backbone.Collection.extend({

    model: Todo,

    localStorage: new Store("todos"),

    done: function() {
      return this.filter(function(todo){ return todo.get('done'); });
    },
```

```
window.TodoView = Backbone.View.extend({

    tagName:  "li",

    template: _.template($('#item-template').html()),

    events: {
      "click .check"              : "toggleDone",
      "dblclick div.todo-text"    : "edit",
      "click span.todo-destroy"   : "clear",
      "keypress .todo-input"      : "updateOnEnter"
    },

    initialize: function() {
      this.model.bind('change', this.render, this);
      this.model.bind('destroy', this.remove, this);
    },

    render: function() {
      $(this.el).html(this.template(this.model.toJSON()));
      this.setText();
      return this;
    },
```

Works like this:



Requires Underscore.js.

Open-source component of DocumentCloud.

Unrestricted custom license...

Current version: 0.9.1.

http://backbonejs.org/

Underscore.js

JavaScript utility belt.

< 4 kb.

~60 functions.

Collections, arrays, helper functions, objects...

Adds support for functional programming.

Usual suspects: map, reduce, filter, select, invoke...

Collections:

- each
- map
- reduce
- find
- filter
- invoke
- pluck
- sortBy/groupBy
- shuffle
- toArray
- ...

Arrays:

- first
- last
- rest
- compact
- union
- intersection
- uniq
- zip
- indexOf
- range
- ...

Array functions work on the *arguments* object.

Functions:

- bind/bindAll
- memoize
- delay
- defer
- throttle
- debounce
- once
- after
- wrap
- compose

Objects:

- keys
- values
- functions
- extend
- defaults
- clone
- isEqual
- isEmpty
- isString/isBoolean/isNumber...
- isNaN/isNull/isUndefined...
- ...

Utility and chaining.

- noConflict
- identity
- times
- mixin
- uniqueId
- escape
- template
- chain
- value

Includes template support.

Can interpolate variables, execute JavaScript.

Modeled after ERB.

Open-source component of DocumentCloud.

Unrestricted custom license...

Current version: 1.3.1.

http://documentcloud.github.com/underscore/

Spine.js

Lightweight framework.

Sensing a theme here?

Written in CoffeeScript.

Doesn't require CoffeeScript though.

About 500 lines of CoffeeScript.

About 2 kb.

Opinionated.

MVC - provides code structure.

Support for classes.

Focus on maintainability, modularity; name spaced.

All about simplicity.

Isn't a UI framework.

Focus on user experience.

Perception of performance.

Asynchronous in nature.

Client updates, server in background.

Enter asynchronous UIs.

Breaks out of the request/ response approach.

Click and wait?

http://alexmaccaw.co.uk/posts/async_ui

Support for mobile apps.

Spine Mobile.

http://spinejs.com/mobile/docs/index

Has an app generator.

Spine.app.

http://spinejs.com/docs/app

Similar to the scripts included with Ruby on Rails.

Hem - JavaScript dependency manager.

http://spinejs.com/docs/hem

Based on CommonJS.

http://spinejs.com/docs/commonjs

Built for book: JavaScript Web Applications.

http://shop.oreilly.com/product/0636920018421.do#

Similar to Backbone.

```
class Task extends Spine.Model
  @configure "Task", "name", "done"

  @extend Spine.Model.Local

  @active: ->
    @select (item) -> !item.done

  @done: ->
    @select (item) -> !!item.done

  @destroyDone: ->
    rec.destroy() for rec in @done()
```

```
class Tasks extends Spine.Controller
  events:
    "change   input[type=checkbox]": "toggle"
    "click    .destroy":             "remove"
    "dblclick .view":                "edit"
    "keypress input[type=text]":     "blurOnEnter"
    "blur     input[type=text]":     "close"

  elements:
    "input[type=text]": "input"

  constructor: ->
    super
    @item.bind("update", @render)
    @item.bind("destroy", @release)

  render: =>
    @replace($("#taskTemplate").tmpl(@item))
```

Focus on good documentation.

Unrestricted custom license...

Current version: 1.0.5.

http://spinejs.com/

Batman.js

Framework for building rich web apps.

Single page apps.

MVC architecture.

Written in CoffeeScript.

Develop in CoffeeScript or JavaScript.

Events - like pub/sub.

Observe function - bind to change event.

Views are HTML.

Models, just as you expect.

Validation, data, etc.

Controllers - again, as you expect.

Inspired by Ruby on Rails.

Also has routes.

@persist - macro, defines storage solution.

Support for local storage, RESTful calls.

Rails based assumptions.

Declarative.

"Designed for developer and designer happiness."

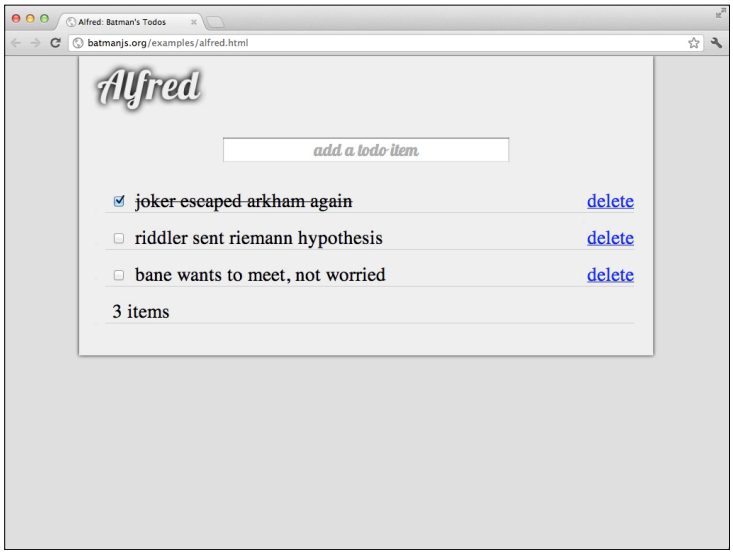
Similar in nature to Spine and Backbone.

For example...

```coffeescript
class Alfred.Todo extends Batman.Model
  @global yes

  @persist Batman.LocalStorage
  @encode 'body', 'isDone'

  body: ''
  isDone: false
```

```coffeescript
class Alfred.TodosController extends Batman.Controller
  index: ->
    @set 'emptyTodo', new Todo

    Todo.load (todos) ->
      if not todos.length
        new Todo(body: 'joker escaped arkham again', isDone: true).save()
        new Todo(body: 'riddler sent riemann hypothesis').save()
        new Todo(body: 'bane wants to meet, not worried').save()

    @render false

  create: =>
    @emptyTodo.save =>
      @set 'emptyTodo', new Todo
```

Starting to look familiar?



Shopify.

http://www.shopify.com/

MIT license.

Current version: 0.8.0.

Fourth alpha.

http://batmanjs.org/

Knockout

Model View View Model.

Smallish.

13 kb minified, gzipped.

Large compared to some!

No dependencies.

Supports mainstream browsers including IE.

Similar concepts: models, changes update view.

Declarative bindings.

Includes templating.

```
<p>First name: <input data-bind="value: firstName" /></p>
<p>Last name: <input data-bind="value: lastName" /></p>

<p>Full name: <strong data-bind="text: fullName"></strong></p>

<button data-bind="click: capitalizeLastName">Go caps</button>
```

```
function AppViewModel() {
    this.firstName = ko.observable("Bert");
    this.lastName = ko.observable("Bertington");

    this.fullName = ko.computed(function() {
        return this.firstName() + " " + this.lastName();
    }, this);

    this.capitalizeLastName = function() {
        var currentVal = this.lastName();
        this.lastName(currentVal.toUpperCase());
    };
}
```

Higher level.

Links UI and model.

Works with jQuery.

MIT License.

Current version: 2.0.0.

http://knockoutjs.com/

Sammy.js

Small.

5.2 kb compressed/gzipped.

Provides structure.

Built on plugins and adapters.

Let's you choose what to include.

Fun is a design goal!

Again, influenced by Ruby and Rails, Sinatra.

Built around REST.

Routes and events.

Routes are built on a verb, a path and a callback.

```
this.get('#/list/:id', function() {
  var list = Lists.get(this.params['id']);
  if (list) {
    this.partial('templates/todolist.template', {
      list: list,
      todos: Todos.filter('listId', list.id)
    }, function(html) {
      $('#todo-list').html(html);
    });
  } else {
    this.notFound();
  }
});
```

Binds to form submit.

Executes callback on route that matches.

Paths are URLs or hashes.

Can use regular expressions.

Events - look just like jQuery event binding.

```
$('.check')
  .live('click', function() {
    var $this = $(this),
      $li = $this.parents('li').toggleClass('done'),
      isDone = $li.is('.done');
    app.trigger('mark' + (isDone ? 'Done' : 'Undone'),
      { id: $li.attr('data-id') });
});
```

Easy to add custom events.

Predefined application lifecycle events.

Plugin system.

Handful included in a public repository.

Designed for modern browsers, mobile WebKit.

Depends on jQuery.

Supports most templating libraries.

MIT license.

Current version: 0.7.1.

http://sammyjs.org/

ClojureScript.

JavaScript "replacement".

Clojure compiler.

Emits JavaScript.

Why?

JavaScript has incredible reach.

It's everywhere: browsers, TVs, web servers.

Write in Clojure...run where JavaScript runs.

Includes name spaces, map, reduce, filter...

Lists, maps, vectors, sets.

Sequences.

Macro support.

Regular expressions, atoms.

Has a REPL.

Aren't analogs for everything.

No runtime evaluation
or compilation.

No Java interop.

Isn't Clojure in JavaScript.

It isn't JavaScript with Clojure syntax.

Works in conjunction with Google's Closure compiler.

Eclipse Public License 1.0.

Currently alpha.

https://github.com/clojure/clojurescript/wiki

CoffeeScript.

Language that "compiles" into JavaScript.

"It's just JavaScript."

The good parts at least.

Alternative syntax.

Significant whitespace.

Love it...or hate it.

Can create objects with indentation.

Similar to YAML.

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

Functions are easier.

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

Less typing!

No need for var.

Compiler handles scope.

Can omit brackets and parentheses on ifs.

Comprehensions largely replace for loops.

```
eat food for food in ['toast', 'cheese', 'wine']
```

Includes ranges.

== compiles into ===
!= into !==

Adds the "existential" operator, the *?*

Like Ruby *nil?*

Returns true, except when it is null or undefined.

Provides a class concept.

And on and on.

Compiled you say...

Output is readable.

Works with any JavaScript library.

Debugging *can* be an issue.

http://ryanflorence.com/2011/2012/case-against-coffeescript/

MIT license.

Current version: 1.2.0.

http://coffeescript.org/

So many more..

SproutCore.

http://sproutcore.com/

Ember (formerly SproutCore 2.0).

https://github.com/emberjs/ember.js

Cappuccino.

http://cappuccino.org/

Angular.

http://angularjs.org/

Bootstrap.

http://twitter.github.com/bootstrap/

Phantom.js

http://www.phantomjs.org/

For example!

Very rich space today...

How do you choose?

Well, what's your problem?

Smaller jQuery?

JavaScript alternative?

Create asynchronous UI?

Many approaches, same basic concepts.

Similar influences.

Which looks right to you?

Differences largely in view.

And how it wires in.

Be aware of licenses.

Most are MIT. Not all!

Many of these
are quite new.

What is your
appetite for “new”?

Browser support.

Varies.

Some leave IE behind.

Will that work for you?

What does it depend on?

Read the docu.

Most of it is quite good.

Lot of options.

Play with them.

Paradox of choice!

Many options.

Exciting time.

Be aware of the alternatives.

Image Credits

- http://www.flickr.com/photos/bobjagendorf/5492860578/

Thanks!

Nathaniel T. Schutta
@ntschutta