

Including jQuery is not an answer!

- Design, techniques and tools for larger JS apps



Trifork A/S, Headquarters
Margrethepladsen 4,
DK-8000 Århus C,
Denmark
info@trifork.com
<http://www.trifork.com>

Karl Krukow
Trifork Geek Night
March 15st, 2011,
Trifork, Aarhus

What is the question, then?

Does your JavaScript look like this?

App Explorer Project Explo...

- public
 - images
 - javascrpts
 - autocomplete
 - images.js
 - utilities.js
 - lang
 - calendar-da.js
 - shadedborder
 - shadedborder.js
 - shortcuts
 - my_shortcuts.js
 - shortcuts.js
 - tiny_mce
 - application.js
 - calendar.js
 - calendar-setup.js
 - class-extend.js
 - control_modal.js
 - controls.js
 - dragdrop.js
 - effects.js
 - prototype.js
 - standard.js
 - plugin_assets
 - README
 - stylesheets
 - 404.html
 - 500.html
 - dispatch.cgi
 - dispatch.fcgi

```
19 </br />
20 <script type="text/javascript">
21   Calendar.setup(
22     {
23       inputField : "entry_event_occurred_at",
24       ifFormat   : "%d-%m-%Y -- %H:%M", // the date format
25       button     : "calendar-trigger",
26       showsTime  : "true"
27     }
28   );</script>
29 <br />
30   <%= label_with_validation 'entry', 'contents', m(:new_note) %><br />
31   <%= text_area 'entry', 'contents', {:class=>"mceEditor", :value => params[:note_text]} %>
32   <br />
33   <div style="float:right">
34     <%= submit_tag m(:save_entry) %> &nbsp;
35   </div>
36
37 </div>
38
39 <% end %>
40 </td>
41
42 </tr>
43 </table>
44
45 </script type="text/javascript" language="javascript">
46   function upd() {
47     selected_people = new Array();
48     var textElement = $('persons_persons');
49     textElement.value.split(',').each(function(e) {
50       e = e.strip().toLowerCase();
51       var id = person_name_2_id[e];
52       if (id != null) {
53         selected_people[id] = id;
54       }
55     });
56     synchronizeImages(selected_people);
57   }
58   upd();
59   Event.observe('persons_persons', 'keyup', upd);
60   var now = new Date();
61 </script>
62
63
64
```

Outline

type filter text

- <%= error_messages_for ... %>
- table
 - width: 100%
 - tr
 - script
 - upd()
 - now
 - div

Progress Console

GeekNightExample script/rails server

Search JsTestDriver

```
krukow:~/Projects/JavaScript/GeekNightExample$
```

What about your server side code?

dagbog_prototype

- app
 - controllers
 - helpers
 - models
 - admittance.rb
 - audit.rb
 - cleanup_service.rb
 - converter.rb
 - entry.rb
 - failed_logins.rb
 - image.rb
 - login.rb
 - patient_entry.rb
 - patient.rb
 - person.rb
 - reporter.rb
 - subject.rb
 - temp_login.rb
 - user_in_admittance.rb
 - user_login.rb
 - user_pref.rb
 - user.rb
 - views
 - pureimage.rb
- config
- data
- db
- doc
- lib
- log
- nbproject
- public
- script
- test
 - dataprovider
 - functional
 - integration
 - unit
 - admittance_test.rb
 - entries_helper_test.rb
 - entry_test.rb
 - failed_logins_test.rb

```
19 class UsersController < ApplicationController
20   #presently cannot destroy Users
21   before_filter :redirect_blinded, :only => [[:destroy]]
22
23   #Autogenerated
24   def index
25     list
26     render :action => 'list'
27   end
28
29   # GETs should be safe (see http://www.w3.org/2001/tag/doc/whenToUseGet
30   verify :method => :post, :only => [ :destroy, :create, :update ],
31   :redirect_to => { :action => :list }
32
33   def list
34     required Operation.READ_USER
35     @allowed = @selected_group.descendants.map {|g| g.id}
36     memberships = Membership.find(:all, :include => { :member => :user },
37     @users = Set.new(memberships.map {|m| m.member.user}).to_a.sort_by {
38   end
39
40   def unblock
41     required Operation.READ_RESTRICTED
42     @blocked_users = User.find(:all, :conditions => "blocked = true")
43   end
44
45   def inactive
46     required Operation.READ_RESTRICTED
47     @inactive_users = (Member.find(:all, :include => [ :user, :memberships
48     m.memberships.length == 0
49     end).map {|m| m.user}.sort_by {|u| u.name}
50   end
51
52   def show
53     required Operation.READ_USER
54     @u = User.find(params[:id])
55   end
56
57   def new
58     required Operation.CREATE_USER
59     @u = User.new
60   end
61
62   def create
63     required Operation.CREATE_USER
```

Progress Console GeekNightExample script/rails server Search JsTestDriver

krukow:~/Projects/JavaScript/GeekNightExample\$

Non-functional requirements for the Server-side

- Maintainability and extensibility
- Technical quality
 - e.g. modularity, reuse, separation of concerns
 - automated testing
 - continuous integration/deployment
 - Tool support (static analysis, compilers, IDEs)
- Productivity
- Performant
- Appropriate architecture and design
- ...

Why so different?

- “Front-end” programming isn't 'real' programming?
- JavaScript isn't a 'real' language?
 - Browsers are impossible...
- That's just the way it is...

The problem is only going to get worse!

- JS apps will get larger and more complex.
- More logic and computation on the client.
- HTML5 and mobile web will require more programming.
- We have to test and maintain these apps.
- We will have harder requirements for performance (e.g. mobile).

Add a number to another number in JavaScript



hallo

I have got a number in my JavaScript variable! Now how do I add another number to it? Please

javascript

tagged

javascript × 18553

asked

a while ago

viewed

some times

latest activity

just now

3 Answers

oldest

newest

votes



You should definitely use jQuery. It's really great and does all things

link | edit | flag

answered 11 minutes ago

 I<3jQuery
1,234 ● 2 ● 13

I agree, jQuery is really the best, it solves all kinds of browser problems and is good, as well – [||sumc0da](#) 8 mins ago

+1 jquery is best quality code ever, if you don't use your a idiot – [Werry_Togan](#) 4 mins ago

add comment

Wanted: Yet another ASP.NET developer. See this and other great job listings at jobs.stackoverflow.com.

Related

[What is the best number?](#)

[How can I use JavaScript to parse some HTML using regex?](#)

[JavaScript: why is my text content getting mangled when I clone nodes? Obviously I must be doing something wrong as jQuery is perfect](#)

[Stupid JavaScript floating point numbers are broken](#)

[How can I extract number from HTML using a regex without ZALGO singing the song that ends the world?](#)

[Is there a jQuery plugin for making an HTML page appear in the browser?](#)

[Where are my legs?](#)



I think there's a jQuery plugin for that. Google for jQuery basic arithmetic plugin.

link | edit | flag

answered 5 minutes ago

 Timothy Goatse
4,321 ● 1 ● 12

yeah, jQuery is definately the way to go – [fishnipples](#) 5 mins ago

I used the jQuery diet plugin and lost 10kg in a week – [jfatty](#) 4 mins ago

add comment




To add numbers together you should use the [+ operator](#), for example:

```
var a= 1;
var b= a+2;
alert(b); // 3
```

link | edit | delete | flag

answered 50 seconds ago

 bobince
some ● ● ●

-1 not enough jQuery – [||sumc0da](#) 30 secs ago

you suck – [Timothy Goatse](#) 3 secs ago

NO

Including jQuery is NOT an answer to these problems.

(Neither is any other js library)

You need to do more.

Improving quality on client side code

- The goal of this talk is to *motivate and help you improve the technical quality* of your JavaScript projects
- Three main points. To improve non-functional quality:
 - you need to understand the language and host APIs.
 - you need design, structure and file-organization as much (or even more) for JavaScript as you do in other languages, e.g. Java
 - there are tools and they can help with quality, productivity, performance.

Agenda

- **JavaScript and larger programs**
 - Problems for larger programs
 - Scope and closures
 - How closures can help in large programs
- **JavaScript Application Design**
 - Namespacing & File organization
 - A Model-View-Controller-Event design pattern
 - Custom events
 - Example illustrated using Ext JS
- **Tools that can help**
 - IDE support, build and deployment
 - Unit testing
 - Acceptance testing/functional testing
 - Continuous integration

Quick Demo of sample project

JavaScript and Larger Programs

JavaScript is easy

- How to program JavaScript:
 - Open web browser and go to Google.
 - Type in what you need (e.g., datepicker JavaScript).
 - You don't have to even read the article, just copy-paste the result into your page.
 - Customize it: don't worry its just like programming Java..
 - Ship it...
- Well... Does this lead to maintainable, consistent, understandable, performant code? (answer is “no”, just in case your are wondering) :)

Pop-quiz

How well do you know JavaScript?

Q1: what does this code do?

```
<div id="mydiv">mydiv</div>
<span id="myspan">myspan</span>
<p id="myp">myp</p>

<script>
var arr = ['mydiv', 'myspan', 'myp'];
for (var i=0, N=arr.length; i<N; i++) {
    var id = arr[i];
    var e = document.getElementById(id);
    e.onclick = function() {
        e.style.display = 'none';
    };
}
</script>
```

Q2: what does this code do?

```
public class ResetOrReturn {
    static int i = 42;

    static int resetOrReturn(int[] arr) {
        if (arr != null) {
            int sum=0;
            for (int i=0;i<arr.length;i++) {
                sum += arr[i];
            }
            return sum;
        } else {
            i = 0;
            return i;
        }
    }

    public static void main(String[] args) {
        ResetOrReturn.resetOrReturn(null);
        System.out.println(ResetOrReturn.i);
    }
}
```

```
var i = 42;

function resetOrReturn(arr) {
    if (arr != null) {
        var sum = 0;
        for (var i=0;i<arr.length;i++){
            sum += arr[i];
        }
        return sum;
    } else {
        i = 0;
        return i;
    }
}

resetOrReturn(null);

alert(i);
```

Q3: What does this code do?

```
Object.create = (function() {  
    function F() {};  
    return function(p) {  
        F.prototype = p;  
        return new F();  
    };  
})();
```

Q4: What is jQuery doing here?

- This is actual code in jQuery-1.4.2, and is run every time that script is loaded...

```
jQuery.support = {};  
//...  
var div = document.createElement("div");  
div.innerHTML = "<input type='radio' name='radiotest' checked='checked' />";  
  
var fragment = document.createDocumentFragment();  
fragment.appendChild( div.firstChild );  
  
jQuery.support.checkClone = fragment.cloneNode(true).cloneNode(true)  
    .lastChild.checked;  
  
jQuery(function() {  
    var div = document.createElement("div");  
    div.style.width = div.style.paddingLeft = "1px";  
  
    document.body.appendChild( div );  
    jQuery.boxModel = jQuery.support.boxModel = div.offsetWidth === 2;  
    document.body.removeChild( div ).style.display = 'none';  
  
    div = null;  
});
```

Key Properties

- Delivered as source code, as opposed to executables
 - Originally intended to be embedded in web pages
- Hosted. Host can expose various objects and methods.
- Dynamically typed
- Dynamic Objects
 - General containers.
- Prototypal inheritance
 - Objects inherit from objects (no classes) (Inspired by Self)
- Functions are first-class citizens, Closures
 - (inspired by Scheme)
- Linkage of modules via global variables

JavaScript as language

- Bad news: JavaScript seems poorly suited for writing large and complex applications.
 - Lacks language support for
 - modules/namespacing/packages
 - Encapsulation
 - Every thing is changeable
 - No static information like types or classes
 - Uses global namespace and has strange scope rules
- Good news:
 - Learn the language well and see ...
 - the features it does have are powerful enough that we can overcome many difficulties.

Explaining Closures!

- Modules in JavaScript are possible via closures.
- Closures and Scope are probably the most misunderstood parts of JavaScript
- Here is a quick explanation...

Local and Global variables

- The only way of creating a new scope in JavaScript is using a function.
 - Hence (almost) all variables not defined inside a function are globally visible and changeable.
 - Even for scripts loaded from different sources.

```
var count = 42;
evens = [];

for (var i=0;i<count;i++) {
    var j = 2*i;
    evens[i] = j;
}

alert(j);
```

```
var count = 42;
evens = [];
function initEvens() {
    for (var i=0;i<count;i++) {
        var j = 2*i;
        evens[i] = j;
    }
}
initEvens();
alert(j);
```

Programs and the global object

- A JavaScript program consists of a list of statements and function declarations.
- Execution of a statement takes place in *an execution context, containing e.g. a scope chain*:
 - a list of objects (each containing properties)
 - defines what variables are available to statements
- There is an object, 'the global object,' that is at the end of the scope chain.
 - the properties of the global object are always available as 'variables' to statements
 - (unless they are shadowed by other scope chain objects)

Variables, properties and scope.

- Variables and properties are quite similar.
 - A global variable is just a property of the global object.
 - A local variable is just a property in another type of object in the scope-chain.
- When looking up the value of a variable, it starts by looking at the first object in the chain, then proceeds until found.
- Each function is associated with a scope-chain.
 - When the function is called a new object is created and added to the end of the scope chain, forming a new chain – containing local vars, params and decls.
 - This new chain is used as scope chain when executing statements in function body.

Example.

```
1      var create_person = function(name){
|
| 3      return {
|         getName: function() {
| | 5         return name;
|         }
|     };
|
2      var p = create_person("Crockford");
|
4      alert(p.getName());
```


Closures

- Functions have access to outer function's lexical environment (local vars, params, decls)
 - If nested function “escapes” then it still has access (we'll say that it “encloses” it's environment)
 - *escape*: using return or by assignment to an outer variable or object property
- A function together with such an “environment” is called a closure.

The module pattern

- Functions can be used for information hiding
 - Private and public state
- The following pattern is called the Module pattern

```
var adam = (function() {  
    var name = "Adam",  
        sins = [];  
    return {  
        getName: function() {return name;},  
        addSin: function(sin) {  
            sins[sins.length] = sin;  
        }  
    };  
})(); ←
```

THIS IS IMPORTANT!

- (First published by Richard Cornford)
- Style guide: parenthesize function!

Agenda

- **JavaScript and larger programs**
 - Problems for larger programs
 - Scope and closures
 - How closures can help in large programs
- **JavaScript Application Design**
 - Namespacing & File organization
 - A Model-View-Controller-Event design
 - Custom events
 - Example illustrated using Ext JS
- **Tools that can help**
 - IDE support, build and deployment
 - Unit testing
 - Acceptance testing/functional testing
 - Continuous integration

Namespaces

Problems of global variables

- Unless the script author does something any reasonably sized program will contain a large set of global variables
 - Typically scattered over many files with often with no particular structure
 - Hard to get an overview of which variables exist
 - Hard to quickly find where a variables is defined (or redefined!!)
 - No notion of public and private functions
- Much greater risk of collision (particularly in mash-up or portal environments)

Solutions

- Module pattern

```
var publicVar = (function() {  
    var x, y, z; // locals  
    // ...  
})();
```

- Those 'public' vars are still all just in the global namespace.
- To avoid collision, “long” or “unique” names could be used “myappNameStoreForEmployees”.
- Also, a technique known as “namespacing” is popular
 - Since objects are just general containers they can be used for organizing the application itself.

Example: Namespacing

```
App = {
  Store: {
    init: function() {
      //...
    }
  },
  UI: {
    init: function() {
      //...
    }
  },
  init: function() {
    App.Store.init();
    App.UI.init();
  }
};
```

- A single global name
- A natural structure for the application
 - Multiple files?

DSL for “namespaces”

- Suppose now you define a module in a separate file which depends on the existence of a number of other modules
 - Each module living in a “namespace” object

```
ns("App.Init");//the module defined in this script

using(App.Init,
      App.Store,App.UI) //module dependencies
.run(function(Init,Store,UI) {
    //Init is App.Init, Store is App.Store, ...

    var privateVar;//module private var

    Init.init = function() {//public API
        //...
    };
    //...
});
```

Naming spaces and file-organization

- One approach to file-organization is to let the directory structures match your “namespace” organization and to put modules in individual files
 - For example: *com.trifork.project.module1*
- Easy to find a module on the file system
- Each file defines a module with a separate concern.



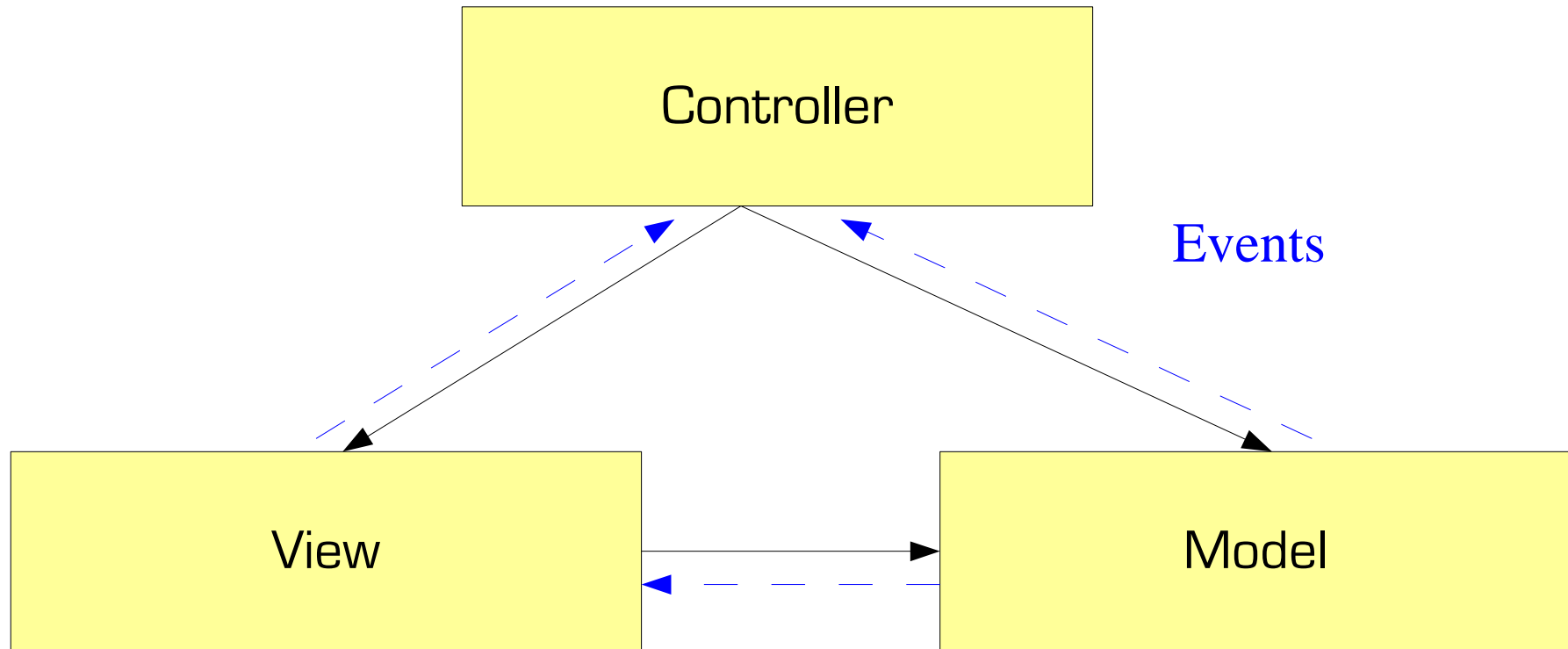
JavaScript Application Design

(or Yet another Model-View-Controller design)

What now?

- We can now split our program into multiple modules, each module being located in a separate file.
 - Using the module pattern (or namespace/using), each module has a public API and potentially private state.
- This is already a big step forward in managing complexity, and is sufficient for many, if not all, programs.
- However,
 - which modules do you want? Is there a common pattern, or is it “each project invents a new way”?
 - how should you name and organize modules and files?

Triangle to the rescue again...



Events/Observer pattern enables loose coupling

Model

- The model
 - Defines domain types with data and methods
 - Provides a central place for accessing application state regarding the domain
 - Broadcasts events when application state changes
 - Provides methods for querying and updating state
- In our example application, a feed reader, the model consists of
 - *com.trifork.experia.Feed* and *com.trifork.experia.Post*
 - A singleton object, *com.trifork.experia.Model*
 - wrapping application state: a number of feeds, each containing a number of posts.

The view

- The view
 - Comprises all the objects managing UI in the application.
 - Often forms a hierarchy/tree of components.
 - Converts user inputs/events to events/actions that make sense at the domain/application level.
 - UI Events can bubble up component tree
- In our example,
 - View consists of several UI components, e.g. a “tree” on the left containing the subscribed feeds and the main panel on the right for reading feed and posts.
 - An object View to which all view events bubble.
 - View components form a tree with the View object at root

The controller

- The controller

- reacts to events originating from the model objects or view objects.
- Updates model objects appropriately on events like user actions
- Updates view objects in reaction to model events to reflect model state

- In our example,

- Controller “connects” the Model object and the View object (since all relevant events bubble to these).
- e.g., when the UI event 'user.newfeed” happens, the model is updated, adding the new feed.
- e.g., when model event 'feed.added' occurs the view is notified and shows the feed.

Other points

- Events carry a 'payload',
 - for example our event 'user.newfeed' carries a *com.trifork.eteria.Feed* object which is the new feed.
- View objects can react to view events too,
 - for example, our view event 'user.selectfeed' both results in the controller updating the model with the currently selected feed
 - **AND** the view reacts by showing the feed in the main panel.

Example: Application Design – Exteria

Agenda

- **JavaScript and larger programs**
 - Problems for larger programs
 - Scope and closures
 - How closures can help in large programs
- **JavaScript Application Design**
 - Namespacing & File organization
 - A Model-View-Controller-Event design
 - Custom events
 - Example illustrated using Ext JS
- **Tools that can help**
 - IDE support, build and deployment
 - Unit testing
 - Acceptance testing/functional testing
 - Continuous integration

Tools that can help
(yes there really are some!)

JavaScript IDEs

- Traditionally JavaScript is edited using simple text editors, or even HTML editors.
- The nature of JavaScript (dynamics, no static types, eval, etc) makes it hard to have “smart” tooling like Eclipse or IDEA
 - However – watch this space
- I use Spket which comes with an understanding of JavaScript and knowledge of several popular JS libraries (jQuery, ExtJS, YUI..)
 - Not near perfect but better than a simple text editor

Analysis

- JSLint is a popular tool that parses your JavaScript and points out errors.
- I run it with every build and it regularly catches errors at “build” time.
 - A good example is the “extra comma” problem that IE6 handles miserably {a: 42, b: 42, c: 42 , }
 - Also catches some scope problems.
- There is much research going into static analysis of JavaScript, for example keep an eye on
 - TAJIS: Type analysis for JavaScript
 - Simon Holm Jensen and Anders Møller and Peter Thiemann
 - <http://www.brics.dk/TAJS/>

Build and deployment

- Splitting your application up into smaller simpler modules and having each module in a file means MANY files
 - Pro managing complexity in large projects,
 - Con: not a good way of distributing JavaScript
- The way you organize files at development time
 - Is not the way you should organize files at runtime.
- Of course, use concatenation, JS-to-JS compilers, gzip compression and HTTP caching.
 - Tooling can help e.g.: YUICompressor, Google Closure Compiler, YSlow, Page Speed

Example

IDE, Static checking
Build, deployment
for Exteria

Testing

- On the server side there has been a movement towards automated testing, both unit and acceptance testing.
 - Techniques like TDD are gaining momentum
- This is often combined with a continuous build/test/integration environment for continuous feedback.
- What about JavaScript?
 - How many of you do some form of automated testing?
 - How many do unit testing?
 - Do you use techniques like “mock” objects?

JavaScript and unit testing

- There are several libraries for unit testing in JavaScript, but it is actually not so easy to find one that easily allows
 - Automated execution (i.e. from the command line)
 - Automated Reporting test outcome in a machine readable form
 - IDE integration
 - Code coverage
- JS-Testdriver
 - <http://code.google.com/p/js-test-driver/>
 - Again not perfect but quite good

Managing Dependencies

- Unit test often require replacing object dependencies with “mock” objects.
- Sinon JS
 - <http://sinonjs.org/>
 - Standalone test spies, stubs and mocks for JavaScript. No dependencies, works with any unit testing framework.
- Support for js-testdriver
- Support for “fake/mock” Ajax requests
- ...

Acceptance/Functional tests

- De-facto standard: Selenium
- Automated
- Easy to integrate with CI servers like hudson
- Selenium 2.0 uses webdriver which enables even more detailed and fine grain automation of browsers than Selenium 1.x.
- API bindings for many languages: Java, C#, Ruby,...
- There is also **Tellurium**
 - <http://code.google.com/p/aost/>

Demo:
Unit testing
Functional testing
Continuous integration

Summary

- Learning JavaScript and DOM apis is useful for debugging, performance, and using the language effectively
 - Learning a library is just a beginning
- We can do design on the client too :)
 - MVC is often useful
 - Libraries can help, e.g. JavaScript MVC and backbone.js for jQuery, ExtJS
 - Custom events help reduce coupling
- Using appropriate tools can help
 - raise our productivity,
 - Web app performance and analysis
 - Quality assurance

References

- Douglas Crockford on JavaScript
 - JavaScript & Advanced JavaScript (and more)
<http://developer.yahoo.com/yui/theater/>
 - JavaScript: The Good Parts
- Namespace/Using on my blog
<http://blog.higher-order.net/2008/02/18/designing-clientserver-web-applications/>
- jQuery number joke
 - <http://www.doxdesk.com/updates/2009.html>
- JavaScript MVC for jQuery
 - <http://www.javascriptmvc.com/>
 - Also: <http://documentcloud.github.com/backbone/>

More references

- JS-Testdriver
 - <http://code.google.com/p/js-test-driver/>
- Spket IDE / Eclipse plugin: <http://spket.com/>
- Selenium: <http://seleniumhq.org/>
- http://www.infoq.com/articles/tellurium_intro
- SinonJS: <http://sinonjs.org/>
- Example Rails project (Exteria)

<http://blog.higher-order.net/files/GeekNightExampleExported.zip>

Additional references

- Performance tooling
 - Google Speed Tracer
 - DynaTrace Ajax Edition
 - YSlow
 - PageSpeed
- Google Closure Compiler
- YUI Compressor
- JSLint
- Nginx
- Steve Souders <http://stevesouders.com/>

Rails plugins

```
gem 'rails', '3.0.3'  
gem 'jammit'  
gem 'jshint_on_rails'  
gem 'selenium-webdriver'  
gem 'selenium-client'  
gem 'test-unit', "2.0"  
gem 'ci_reporter'  
gem 'sqlite3-ruby', :require => 'sqlite3'  
gem "mongrel", ">= 1.2.0.pre2"
```

- *And Ruby 1.9.2*
- *There are equivalent tools for Java and probably .NET :)*