

Thomas Fuchs

Adventures in JavaScript testing

wollzelle
mollzelle

Thomas Fuchs

wol||ze||e

script.aculo.us

mir.aculo.us



traditional
javascript „testing“

every iTunes Music Store customer



<http://www.apple.com>

There was a problem retrieving the XML data:

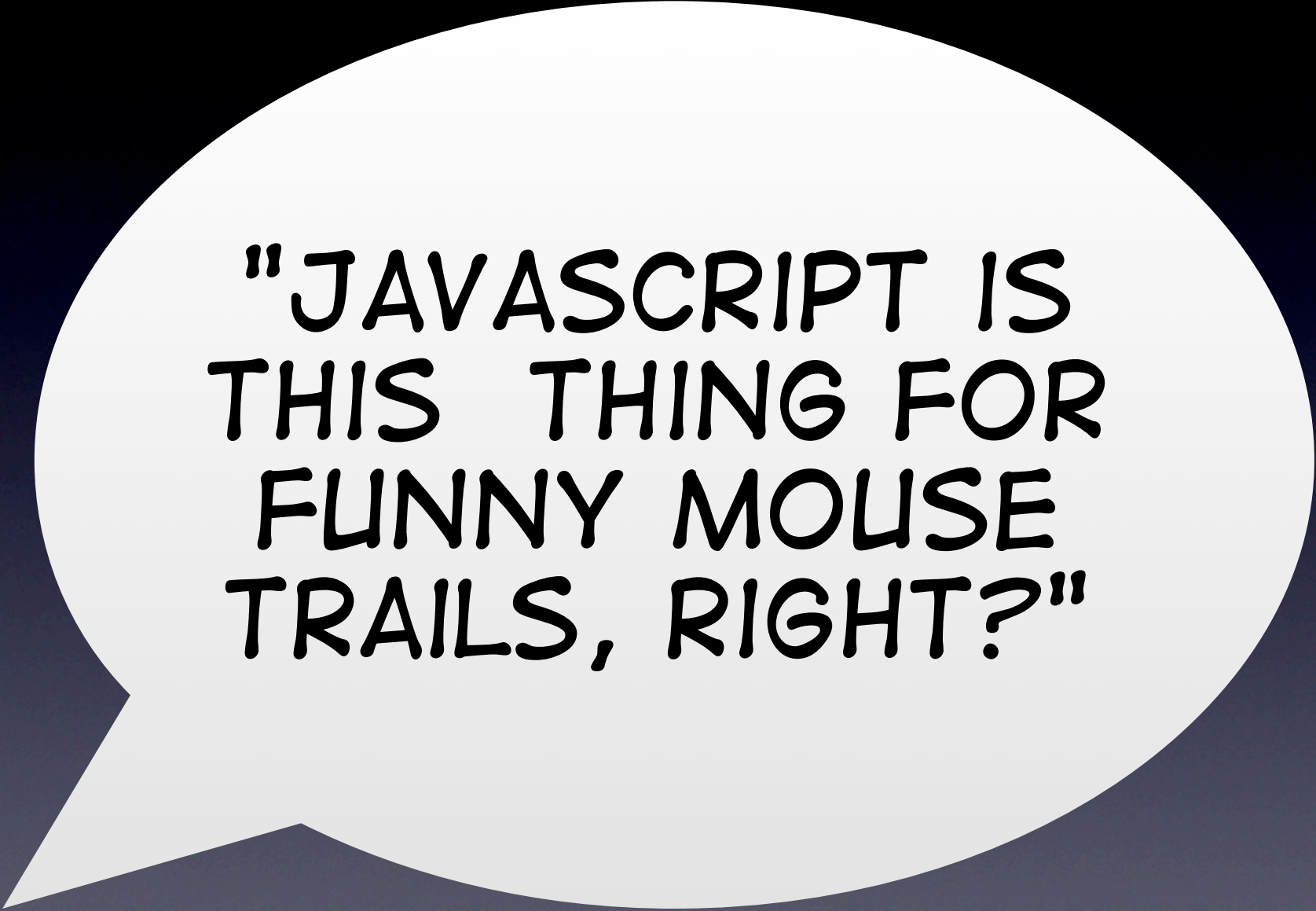
undefined

OK




```
alert("please use me only  
where appropriate!");
```

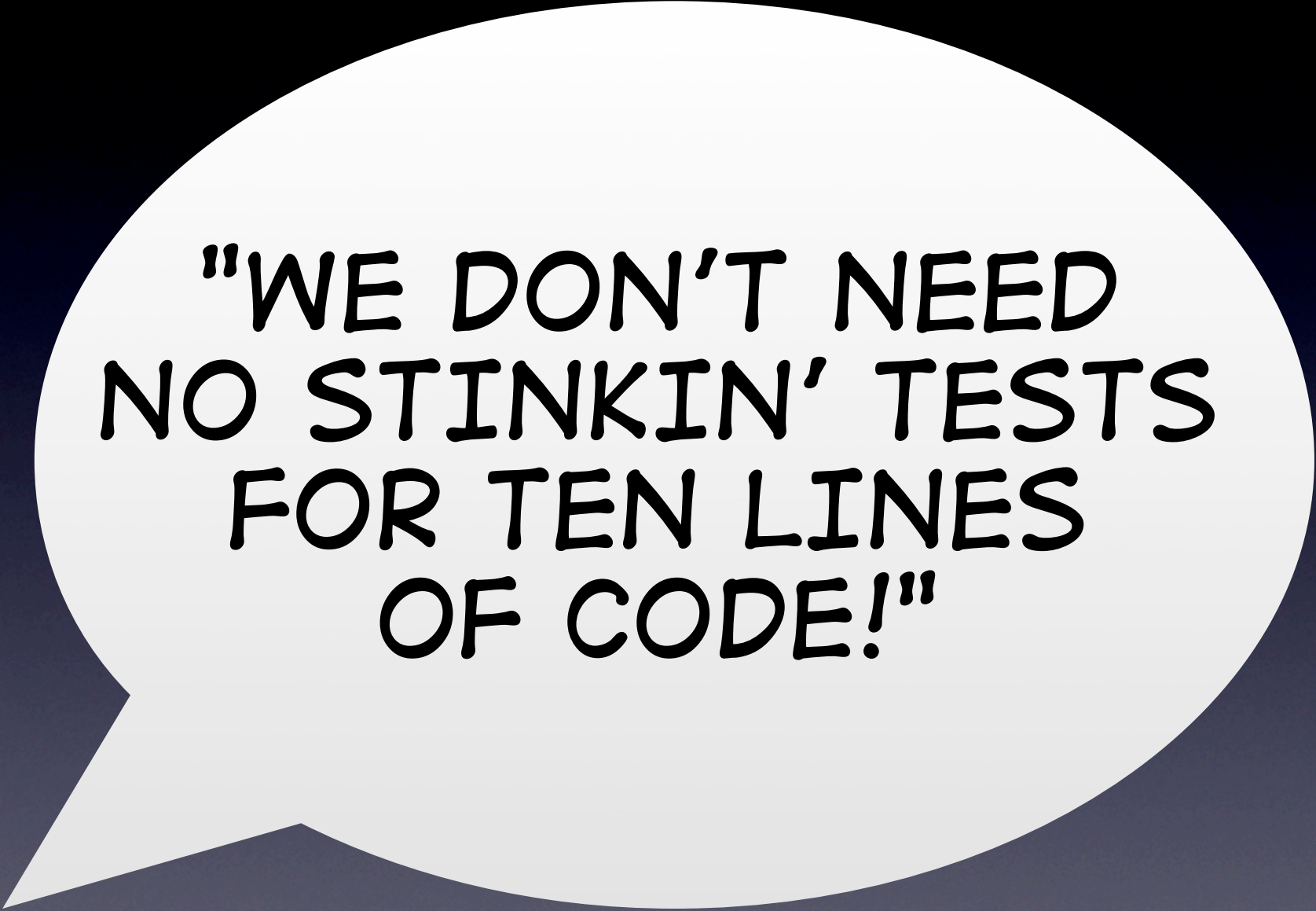
why?



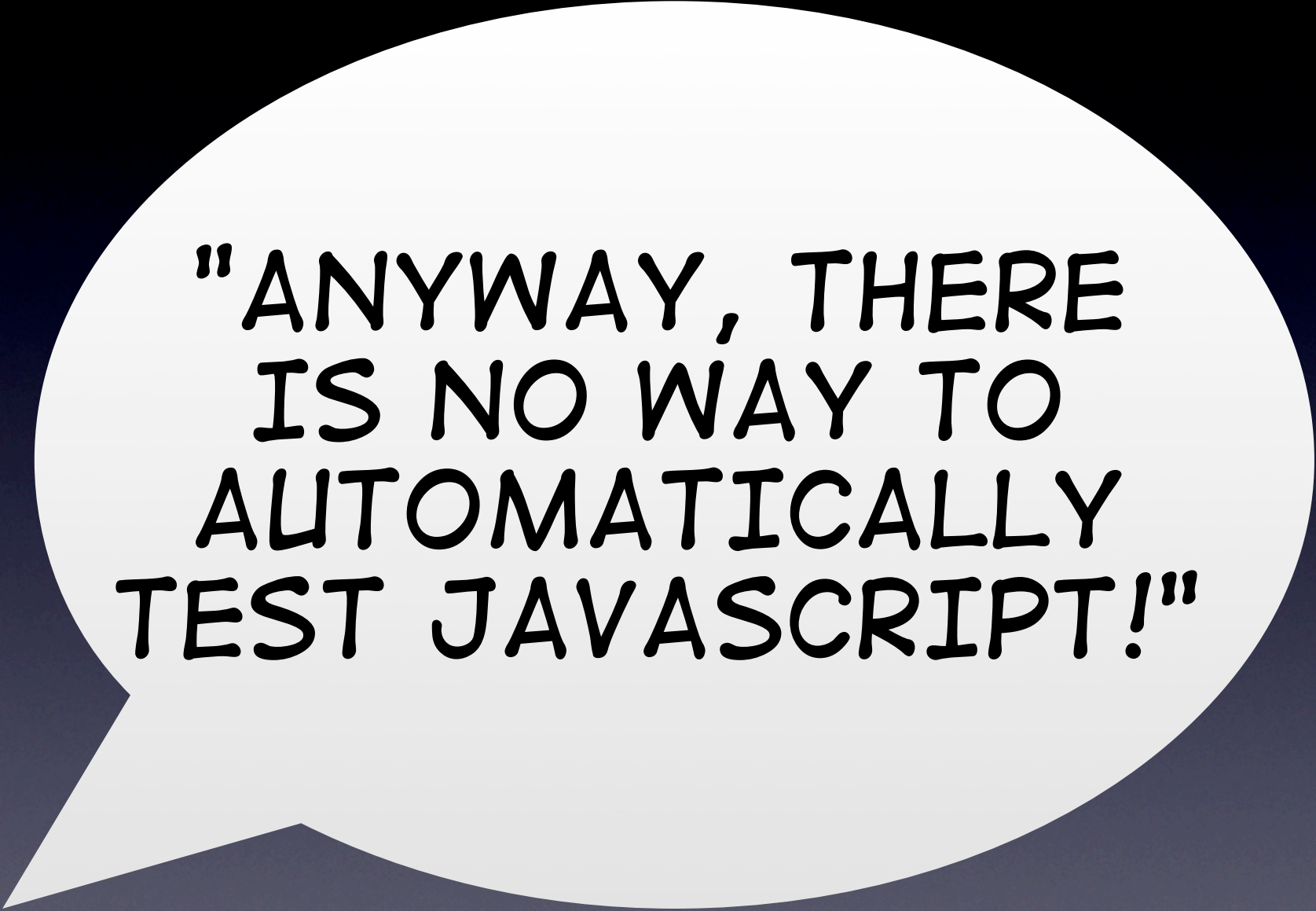
"JAVASCRIPT IS
THIS THING FOR
FUNNY MOUSE
TRAILS, RIGHT?"



**"BUT YOU
CAN'T DEBUG
JAVASCRIPT!"**



**"WE DON'T NEED
NO STINKIN' TESTS
FOR TEN LINES
OF CODE!"**



**"ANYWAY, THERE
IS NO WAY TO
AUTOMATICALLY
TEST JAVASCRIPT!"**

all of this is
pure FUD

unit testing javascript

script.aculo.us Builder unit test file

Tests for Builder.

6 tests, 331 assertions, 0 failures, 0 errors

Status	Test	Message
passed	testBuilderBasics	22 assertions, 0 failures, 0 errors
passed	testBuilderAllXHTMLTags	284 assertions, 0 failures, 0 errors
passed	testBuilderOptionTag	6 assertions, 0 failures, 0 errors
passed	testBuilderContatenation	15 assertions, 0 failures, 0 errors
passed	testBuilderComplexExample	2 assertions, 0 failures, 0 errors
passed	testBuilderShortcuts	2 assertions, 0 failures, 0 errors

```

<html>
<head>
  <title>script.aculo.us Unit test file</title>
  <script src="../../lib/prototype.js" type="text/javascript"></script>
  <script src="../../src/scriptaculous.js" type="text/javascript"></script>
  <script src="../../src/unittest.js" type="text/javascript"></script>
  <link rel="stylesheet" href="../../test.css" type="text/css" />
</head>
<body>
  <!-- Log output -->
  <div id="testlog"> </div>
  <!-- Tests follow -->
  <script type="text/javascript" language="javascript" charset="utf-8">
  // 
    new Test.Unit.Runner({
      setup: function() {
        // setup runs before each test
      },
      teardown: function() {
        // teardown runs after each test
      },
      testBuilderBasics: function() { with(this) {
        var element = Builder.node('div');
        assertEquals('DIV', element.nodeName);
      }}
    });
  // ]]&gt;
&lt;/script&gt;
&lt;/body&gt;
&lt;/html&gt;
</pre>
</div>
<div data-bbox="587 682 904 917" data-label="Text">
<p>Borrows from<br/>Test::Unit</p>
</div>
```

```
<html>
<head>
  <title>script.aculo.us Unit test file</title>
  <script src="../../lib/prototype.js" type="text/javascript"></script>
  <script src="../../src/scriptaculous.js" type="text/javascript"></script>
  <script src="../../src/unittest.js" type="text/javascript"></script>
  <link rel="stylesheet" href="test.css" type="text/css" />
</head>
<body>
<!-- Log output -->
<div id="testlog"> </div>
<!-- Tests follow -->
<script type="text/javascript" language="javascript" charset="utf-8">
// 
  new Test.Unit.Runner({
    setup: function() {
      // setup runs before each test
    },
    teardown: function() {
      // teardown runs after each test
    },
    testBuilderBasics: function() { with(this) {
      var element = Builder.node('div');
      assertEquals('DIV', element.nodeName);
    }}
  });
// ]]&gt;
&lt;/script&gt;
&lt;/body&gt;
&lt;/html&gt;</pre></div><div data-bbox="564 251 909 365" data-label="Text"><p>include unittest.js<br/>(requires Prototype)</p></div>
```



```

<html>
<head>
  <title>script.aculo.us Unit test file</title>
  <script src="../../lib/prototype.js" type="text/javascript"></script>
  <script src="../../src/scriptaculous.js" type="text/javascript"></script>
  <script src="../../src/unittest.js" type="text/javascript"></script>
  <link rel="stylesheet" href="../../test.css" type="text/css" />
</head>
<body>
  <!-- Log output -->
  <div id="testlog"> </div>
  <!-- Tests follow -->
  <script type="text/javascript" language="javascript" charset="utf-8">
  // 
    new Test.Unit.Runner({
      setup: function() {
        // setup runs before each test
      },
      teardown: function() {
        // teardown runs after each test
      },
      testBuilderBasics: function() { with(this) {
        var element = Builder.node('div');
        assertEquals('DIV', element.nodeName);
      }}
    });
  // ]]&gt;
&lt;/script&gt;
&lt;/body&gt;
&lt;/html&gt;
</pre>
</div>
<div data-bbox="557 439 990 614" data-label="Text">
<p>a DIV is required for<br/>containing the test results<br/>(defaults to "testlog")</p>
</div>
```

```

<html>
<head>
  <title>script.aculo.us Unit test file</title>
  <script src="../../lib/prototype.js" type="text/javascript"></script>
  <script src="../../src/scriptaculous.js" type="text/javascript"></script>
  <script src="../../src/unittest.js" type="text/javascript"></script>
  <link rel="stylesheet" href="../../test.css" type="text/css" />
</head>
<body>
  <!-- Log output -->
  <div id="testlog"> </div>
  <!-- Tests follow -->
  <script type="text/javascript" language="javascript" charset="utf-8">
  // 
    new Test.Unit.Runner({
      setup: function() {
        // setup runs before each test
      },
      teardown: function() {
        // teardown runs after each test
      },
      testBuilderBasics: function() { with(this) {
        var element = Builder.node('div');
        assertEquals('DIV', element.nodeName);
      }}
    });
  // ]]&gt;
&lt;/script&gt;
&lt;/body&gt;
&lt;/html&gt;
</pre>
</div>
<div data-bbox="560 514 976 619" data-label="Text">
<p>create a new instance of<br/>Test.Unit.Runner...</p>
</div>
```

```

<html>
<head>
  <title>script.aculo.us Unit test file</title>
  <script src="../../lib/prototype.js" type="text/javascript"></script>
  <script src="../../src/scriptaculous.js" type="text/javascript"></script>
  <script src="../../src/unittest.js" type="text/javascript"></script>
  <link rel="stylesheet" href="../../test.css" type="text/css" />
</head>
<body>
  <!-- Log output -->
  <div id="testlog"> </div>
  <!-- Tests follow -->
  <script type="text/javascript" language="javascript" charset="utf-8">
  // 
    new Test.Unit.Runner({
      setup: function() {
        // setup runs before each test
      },
      teardown: function() {
        // teardown runs after each test
      },
      testBuilderBasics: function() { with(this) {
        var element = Builder.node('div');
        assertEquals('DIV', element.nodeName);
      }}
    });
  // ]]&gt;
&lt;/script&gt;
&lt;/body&gt;
&lt;/html&gt;
</pre>
</div>
<div data-bbox="580 483 953 649" data-label="Text">
<p>...this will excute all<br/>methods starting with<br/>"test"</p>
</div>
<div data-bbox="422 600 668 698" data-label="Image">
<img alt="A white arrow pointing from the text '...this will excute all methods starting with &quot;test&quot;' to the 'testBuilderBasics' method in the code."/>
</div>
```



```
<html>
<head>
  <title>script.aculo.us Unit test file</title>
  <script src="../../lib/prototype.js" type="text/javascript"></script>
  <script src="../../src/scriptaculous.js" type="text/javascript"></script>
  <script src="../../src/unittest.js" type="text/javascript"></script>
  <link rel="stylesheet" href="../test.css" type="text/css" />
</head>
<body>
  <!-- Log output -->
  <div id="testlog"> </div>
  <!-- Tests follow -->
  <script type="text/javascript" language="javascript" charset="utf-8">
// 
  new Test.Unit.Runner({
    setup: function() {
      // setup runs before each test
    },
    teardown: function() {
      // teardown runs after each test
    },
    testBuilderBasics: function() { with(this) {
      var element = Builder.node('div');
      assertEquals('DIV', element.nodeName);
    }}
  });
// ]]&gt;
&lt;/script&gt;
&lt;/body&gt;
&lt;/html&gt;</pre></div><div data-bbox="564 541 770 594" data-label="Text"><p>assert away!</p><img alt="A white arrow pointing from the text 'assert away!' to the 'assertEquals' method call in the code." data-bbox="280 600 670 750"/></div>
```



```

<html>
<head>
  <title>script.aculo.us Unit test file</title>
  <script src="../../lib/prototype.js" type="text/javascript"></script>
  <script src="../../src/scriptaculous.js" type="text/javascript"></script>
  <script src="../../src/unittest.js" type="text/javascript"></script>
  <link rel="stylesheet" href="../../test.css" type="text/css" />
</head>
<body>
  <!-- Log output -->
  <div id="testlog"> </div>
  <!-- Tests follow -->
  <script type="text/javascript" language="javascript" charset="utf-8">
  // 
    new Test.Unit.Runner({
      setup: function() {
        // setup runs before each test
      },
      teardown: function() {
        // teardown runs after each test
      },
      testBuilderBasics: function() { with(this) {
        var element = Builder.node('div');
        assertEquals('DIV', element.nodeName);
      }}
    });
  // ]]&gt;
&lt;/script&gt;
&lt;/body&gt;
&lt;/html&gt;
</pre>
</div>
<div data-bbox="564 451 967 625" data-label="Text">
<p>(syntax strangeness<br/>for more convenience<br/>when calling assertions)</p>
</div>
<div data-bbox="557 621 638 688" data-label="Image">
<img alt="A white arrow pointing from the text '(syntax strangeness for more convenience when calling assertions)' to the 'with(this)' syntax in the JavaScript code."/>
</div>
```

```
assert(expression)
```

```
assert(true) => PASS  
assert(false) => FAIL  
assert(2*2==4) => PASS
```



```
assertEqual(expected, actual)
```

```
assertEqual('a', 'a') => PASS
```

```
assertEqual('a', 'b') => FAIL
```

```
assertEqual(1, 1) => PASS
```

```
assertEqual(1, '1') => PASS
```



does not compare type

```
assertEnumEqual(expected, actual)
```

```
assertEnumEqual([1,2], [1,2]) => PASS
```

```
assertEqual([1,2], [1,2]) => FAIL
```



assertEqual does not
compare enum contents


```
assertNotEqual(expected, actual)
```

```
assertNotEqual('a', 'a') => FAIL
```

```
assertNotEqual('a', 'b') => PASS
```

```
assertNotEqual(1, 1) => FAIL
```

```
assertNotEqual(1, '1') => FAIL
```



does not compare type


```
assertMatch(expected, actual)
```

```
assertMatch(/a/, 'a') => PASS
```

```
assertMatch(/a/, 'bab') => PASS
```

```
assertMatch(/a/, 'b') => FAIL
```

```
assertMatch(/^moo$/, 'moo') => PASS
```

```
assertIdentical(expected, actual)
```

```
assertIdentical('a', 'a') => PASS
```

```
assertIdentical(1, '1') => FAIL
```



compares type


```
assertNotIdentical(expected, actual)
```

```
assertNotIdentical('a', 'a') => FAIL  
assertNotIdentical(1, '1') => PASS
```



compares type

```
assertType(expected, actual)
```

```
assertType(String, 'a') => PASS  
assertType(Number, 1) => PASS  
assertType(Array, [1, 2]) => PASS
```

checks for a specific
constructor


```
assertRaise(exceptionName, method)
```

```
assertRaise('ElementDoesNotExistError',
    function(){
        new Effect.Opacity('invalid-element')
    }
);
```

related effects.js code

```
if(!this.element) throw(Effect._elementDoesNotExistError);
```

```
var Effect = {  
  _elementDoesNotExistError: {  
    name: 'ElementDoesNotExistError',  
    message: 'The specified DOM element does not exist, but is required for this effect to operate'-  
  },  
}
```

`assertRespondsTo(method, obj)`

```
var someObject = {  
  doSomething: function(){  
    // ...  
  }  
}
```

```
assertRespondsTo('doSomething', someObject);
```



```
assertVisible(element)
```

```
assertNotVisible(element)
```

also checks if any parent
elements are hidden and
fails/passes on that

```
info(message)
```

Displays arbitrary
messages in the "Message"
column of the test result


```
assertXYZ(params, message)
```

```
assertNotVisible('testcss2',  
  "Due to a Safari bug, this test fails in Safari.");
```


failed	testAssertVisible	27 assertions, 1 failures, 0 errors Failure: 'testcss2' was not hidden and didn't have a hidden parent either. <u>Due to a Safari bug, this test fails in Safari.</u>
--------	-------------------	---



**"WELL, WHAT ABOUT
FUNCTIONS THAT
RUN ASYNCHRONOUSLY?"**

`wait(milliseconds, method)`

```
testInspect: function() { with(this) {  
  var e1 = new Effect.Opacity('sandbox', {from:1.0,to:0.5,duration:0.5});  
  assertEquals(0, e1.inspect().indexOf('#<Effect:'));  
  assert(e1.inspect().indexOf('idle')>0);  
  wait(1000, function() {  
    assert(e1.inspect().indexOf('finished')>0);  
  });  
}},
```



should be last statement in the
test (but can be nested)

Benchmarking

```
benchmark(function(){  
  Element.childrenWithClassName("Container", "firstClass")  
}, 1000, 'Element.childrenWithClassName');
```

passed

testElementChildrenWithClassName

10 assertions, 0 failures, 0 errors

Info: Element.childrenWithClassName finished
1000 iterations in 1.254s



**"WELL, NICE. BUT IT'S
TEDIOUS TO RUN ALL THOSE
TESTS MANUALLY!"**

rake to the rescue

```
rake test:javascripts
```



```
$ rake test:javascripts  
(in /Users/thomasfuchs/Web/fluxiom-dev)  
/test/javascript/fluxiom_test.html on Safari: SUCCESS  
/test/javascript/fluxiom_test.html on Firefox: SUCCESS  
Skipping Internet Explorer, not supported on this OS  
Skipping Konqueror, not supported on this OS
```

JavaScript unit test file

http://localhost:4711/test/javascript/foobar_test.html?resultsURL=http://local Google

JavaScript unit test file

This file tests foobar.js.

1 tests, 1 assertions, 0 failures, 0 errors

Status	Test	Message
passed	testTruth	1 assertions, 0 failures, 0 errors

Menü anzeigen

JavaScript unit test file

http://localhost:4711/test/javascript/foobar_test. G

JavaScript unit test file

This file tests foobar.js.

1 tests, 1 assertions, 0 failures, 0 errors

Status	Test	Message
passed	testTruth	1 assertions, 0 failures, 0 errors

Fertig

javascript_test plugin

rake test:javascripts

1. launches
web server

WEBrick

lists results
SUCCESS
FAILURE
ERROR

2. controls
browsers



3. run tests

```
$ script/generate javascript_test muhaha  
exists    test/javascript  
create    test/javascript/muhaha_test.html
```


just
add your
tests

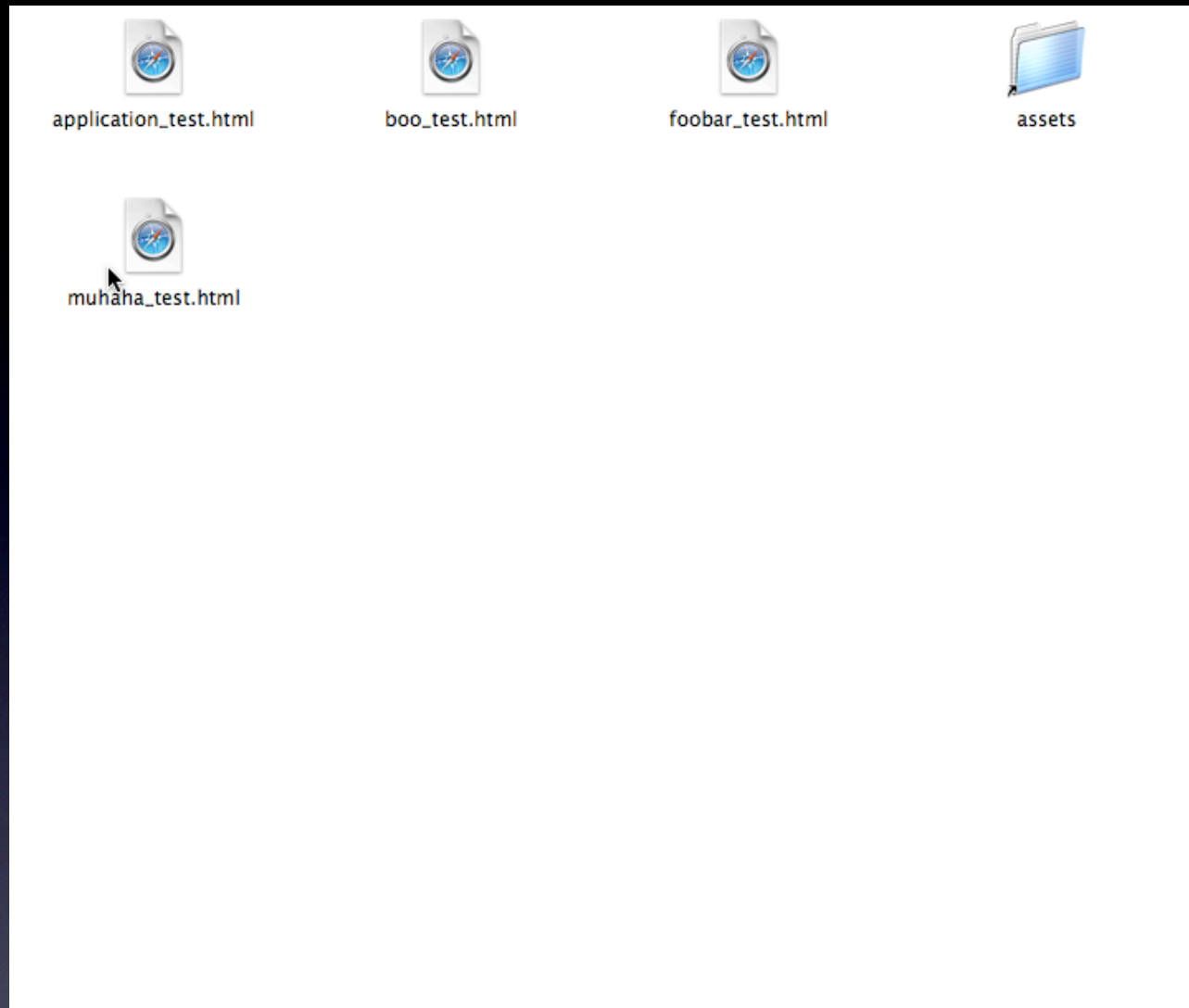
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
  <title>JavaScript unit test file</title>
  <meta http-equiv="content-type" content="text/html; charset=utf-8" />
  <script src="assets/prototype.js" type="text/javascript"></script>
  <script src="assets/unittest.js" type="text/javascript"></script>

  <script src="../../public/javascripts/muhaha.js" type="text/javascript"></script>

  <link rel="stylesheet" href="assets/unittest.css" type="text/css" />
</head>
<body>
  <div id="content">
    <div id="header">
      <h1>JavaScript unit test file</h1>
      <p>
        This file tests <strong>muhaha.js</strong>.
      </p>
    </div>

    <!-- Log output -->
    <div id="testlog"> </div>
  </div>

  <script type="text/javascript">
    // <![CDATA[
    new Test.Unit.Runner({
      // place this with your real tests
      setup: function() {
      },
      teardown: function() {
      },
      testTruth: function() { with(this) {
        assert(true);
      }}
    }, "testlog");
    // ]]>
  </script>
</body>
</html>
```

(symlink required:
app/test/javascript/assets to
app/vendor/plugins/javascript_test/assets)

Browser definitions

(add more!)

```
class FirefoxBrowser < Browser
  def initialize(path='c:\Program Files\Mozilla Firefox\firefox.exe')
    @path = path
  end

  def visit(url)
    applescript('tell application "Firefox" to Get URL "' + url + '"') if macos?
    system("#{@path} #{url}") if windows?
    system("firefox #{url}") if linux?
  end

  def to_s
    "Firefox"
  end
end
```

```
$ script/plugin install http://dev.rubyonrails.org/svn/rails/plugins/javascript\_test
```


...and what about BDD?



```
Test.context("BDD-style testing",{  
  'should provide extensions to tie in isSomething object methods': function(){  
    Object.extend(testObj, Test.BDDMethods);  
    testObj.shouldBe('nice');  
    testObj.shouldNotBe('broken');  
  },  
  'should automatically add extensions to strings': function(){  
    'a'.shouldEqual('a');  
    'a'.shouldNotEqual('b');  
    'a'.shouldNotBeNull();  
    'a'.shouldBeA(String);  
    var aString = 'boo!';  
    aString.shouldEqual('boo!');  
    aString.shouldBeA(String);  
    aString.shouldNotBeA(Number);  
  }  
});
```


**Borrows from
RSpec**

specifications



```
Test.context("BDD-style testing",{  
  'should provide extensions to tie in isSomething object methods': function(){  
    Object.extend(testObj, Test.BDDMethods);  
      
    testObj.shouldBe('nice');  
    testObj.shouldNotBe('broken');  
  },  
    
  'should automatically add extensions to strings': function(){  
    'a'.shouldEqual('a');  
    'a'.shouldNotEqual('b');  
    'a'.shouldNotBeNull();  
    'a'.shouldBeA(String);  
      
    var aString = 'boo!';  
    aString.shouldEqual('boo!');  
    aString.shouldBeA(String);  
    aString.shouldNotBeA(Number);  
  }  
});
```


```
Test.context("BDD-style testing",{  
  'should provide extensions to tie in isSomething object methods': function(){  
    Object.extend(testObj, Test.BDDMethods);  
    testObj.shouldBe('nice');  
    testObj.shouldNotBe('broken');  
  },  
  'should automatically add extensions to strings': function(){  
    'a'.shouldEqual('a');  
    'a'.shouldNotEqual('b');  
    'a'.shouldNotBeNull();  
    'a'.shouldBeA(String);  
    var aString = 'boo!';  
    aString.shouldEqual('boo!');  
    aString.shouldBeA(String);  
    aString.shouldNotBeA(Number);  
  }  
});
```



add BDD methods
to any object


```
Test.context("BDD-style testing",{  
  'should provide extensions to tie in isSomething object methods': function(){  
    Object.extend(testObj, Test.BDDMethods);  
    testObj.shouldBe('nice');  
    testObj.shouldNotBe('broken');  
  },  
  'should automatically add extensions to strings': function(){  
    'a'.shouldEqual('a');  
    'a'.shouldNotEqual('b');  
    'a'.shouldNotBeNull();  
    'a'.shouldBeA(String);  
  
    var aString = 'boo!';  
    aString.shouldEqual('boo!');  
    aString.shouldBeA(String);  
    aString.shouldNotBeA(Number);  
  }  
});
```

for isSomething()
methods, that
return a boolean




```
Test.context("BDD-style testing",{  
  'should provide extensions to tie in isSomething object methods': function(){  
    Object.extend(testObj, Test.BDDMethods);  
    testObj.shouldBe('nice');  
    testObj.shouldNotBe('broken');  
  },  
  'should automatically add extensions to strings': function(){  
    'a'.shouldEqual('a');  
    'a'.shouldNotEqual('b');  
    'a'.shouldNotBeNull();  
    'a'.shouldBeA(String);  
    var aString = 'boo!';  
    aString.shouldEqual('boo!');  
    aString.shouldBeA(String);  
    aString.shouldNotBeA(Number);  
  }  
});
```

should* assertions,
directly on objects



```
Test.context("BDD-style testing",{  
  'should provide extensions to tie in isSomething object methods': function(){  
    Object.extend(testObj, Test.BDDMethods);  
    testObj.shouldBe('nice');  
    testObj.shouldNotBe('broken');  
  },  
  'should automatically add extensions to strings': function(){  
    'a'.shouldEqual('a');  
    'a'.shouldNotEqual('b');  
    'a'.shouldNotBeNull();  
    'a'.shouldBeA(String);  
    var aString = 'boo!';  
    aString.shouldEqual('boo!');  
    aString.shouldBeA(String);  
    aString.shouldNotBeA(Number);  
  }  
});
```

↑
("with" ugliness no longer
required)

script.aculo.us Unit test file

BDD-style testing: 10 tests, 47 assertions, 0 failures, 0 errors

Status	Test	Message
passed	-> should run setup before each specification	2 assertions, 0 failures, 0 errors
passed	-> should run teardown after each specification	1 assertions, 0 failures, 0 errors
passed	-> should provide extensions to tie in isSomething object methods	2 assertions, 0 failures, 0 errors
passed	-> should automatically add extensions to strings	7 assertions, 0 failures, 0 errors
passed	-> should automatically add extensions to numbers	4 assertions, 0 failures, 0 errors
passed	-> should automatically add extensions to arrays	3 assertions, 0 failures, 0 errors
passed	-> should support the eval() method	1 assertions, 0 failures, 0 errors
passed	-> should support equality assertion	5 assertions, 0 failures, 0 errors
passed	-> should provide all assertions	19 assertions, 0 failures, 0 errors
passed	-> should support deferred execution	3 assertions, 0 failures, 0 errors

Mapping

```
var METHODMAP = {  
  shouldEqual: 'assertEqual',  
  shouldNotEqual: 'assertNotEqual',  
  shouldEqualEnum: 'assertEnumEqual',  
  shouldBeA: 'assertType',  
  shouldNotBeA: 'assertNotOfType',  
  shouldBeAn: 'assertType',  
  shouldNotBeAn: 'assertNotOfType',  
  shouldBeNull: 'assertNull',  
  shouldNotBeNull: 'assertNotNull',  
  shouldBe: 'assertReturnsTrue',  
  shouldNotBe: 'assertReturnsFalse',  
};
```

Mixed environment

```
'should automatically add extensions to arrays': function(){  
  ['a'].shouldNotBeA(String);  
  [1,2,3].shouldBeAn(Array);  
  [1,2,3].shouldEqualEnum([1,2,3]);  
},  
  
'should support the eval() method': function(){  
  eval('2*2').shouldEqual(4);  
},  
  
'should support equality assertion': function(){  
  assertEquals(1, 1);  
  assertEquals('a', 'a');  
  assertEquals(1, '1');  
  
  var x = 1;  
  var y = 1;  
  assertEquals(1, x);  
  assertEquals(x, y);  
},
```

can also use
"normal" assertions



...and if something
breaks?



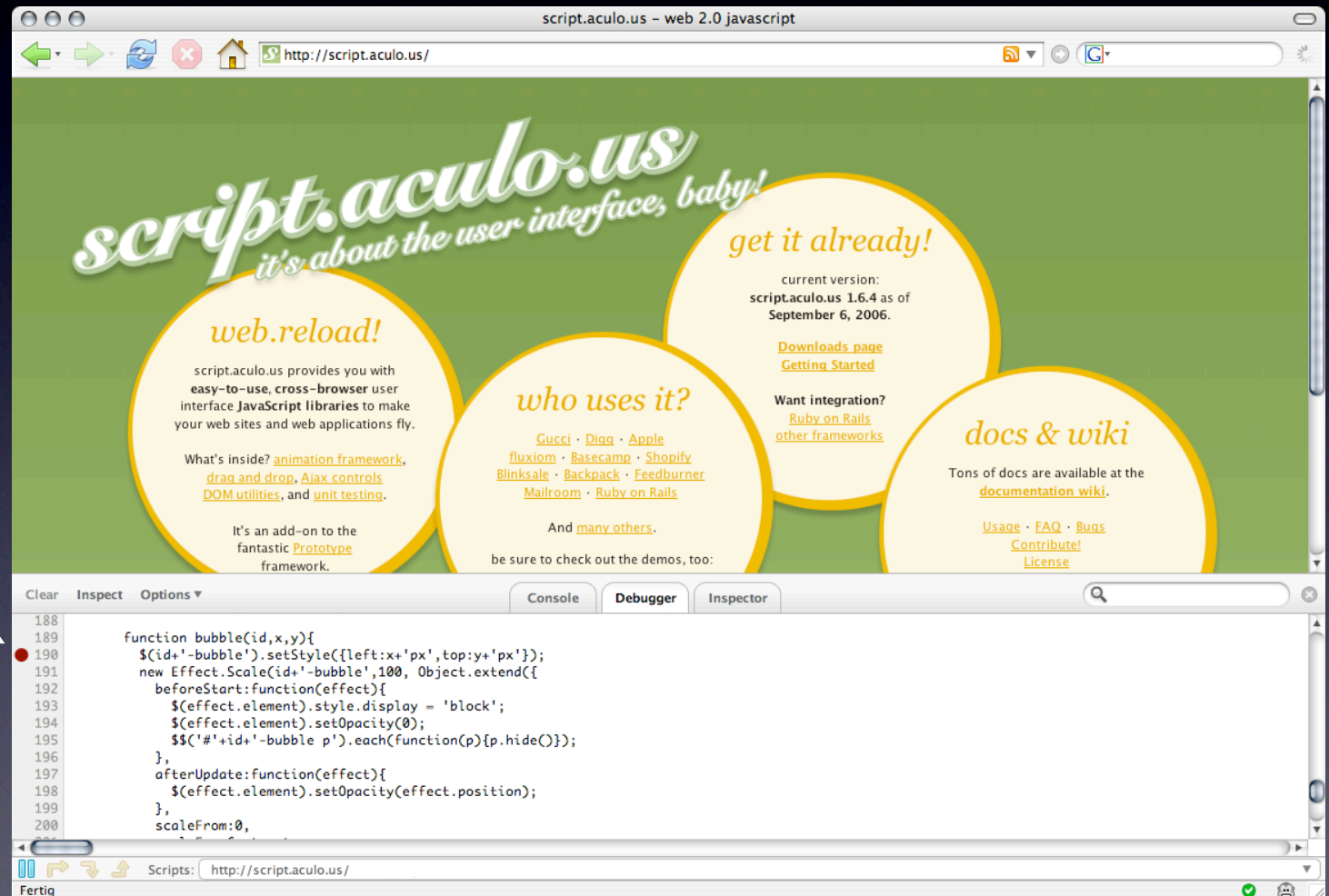
Firebug



sliced bread

Firebug

Debugger



Firebug

know
your
Ajax




script.aculo.us - web 2.0 javascript demos


http://demo.script.aculo.us/shop

silly easy shopping

Drag products to the cart to fill it:




your cart:

 T-Shirt (1)

Drop items here to remove them from the cart.

© 2005 thomas fuchs [license](#) [validate](#) [mir.aculo.us](#)

rails 0.13 demos
script.aculo.us helpers
[Autocompleting text fields \(basic\)](#)
[Autocompleting text fields \(customized\)](#)
[Shopping cart](#)
[Sortable elements](#)
New AJAX features
[Error handling](#)
[Update element helper](#)



Rails is a full-stack, open-source web framework in Ruby for writing real-world applications with joy and less code than most frameworks spend doing XML sit-ups. That quite sums it up. So, if you're still working late hours writing definitions of what's in your database because the framework you use works against and not for you, you should try Rails. [Find out more about Rails](#)

script.aculo.us demo site version 2005/07/04

Clear Inspect Options ▾ Console Debugger Inspector

POST http://demo.script.aculo.us/shop/add prototype.js (line 230)

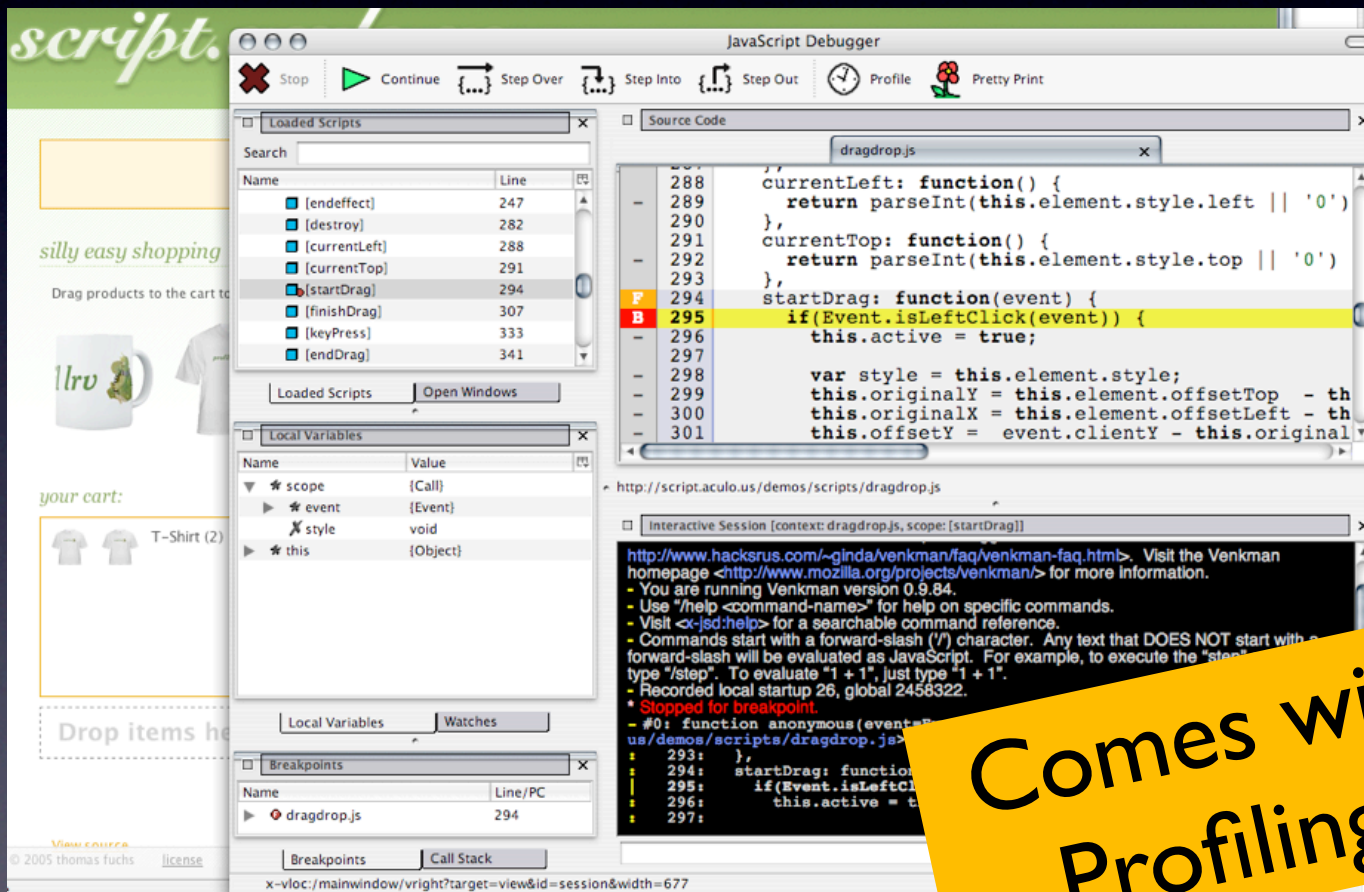
Post Response Headers

```
<div>
  
  <script type="text/javascript">new Draggable('item_2_0', {revert:true})</script>

  <span class="title">T-Shirt (1)</span>
</div>
```

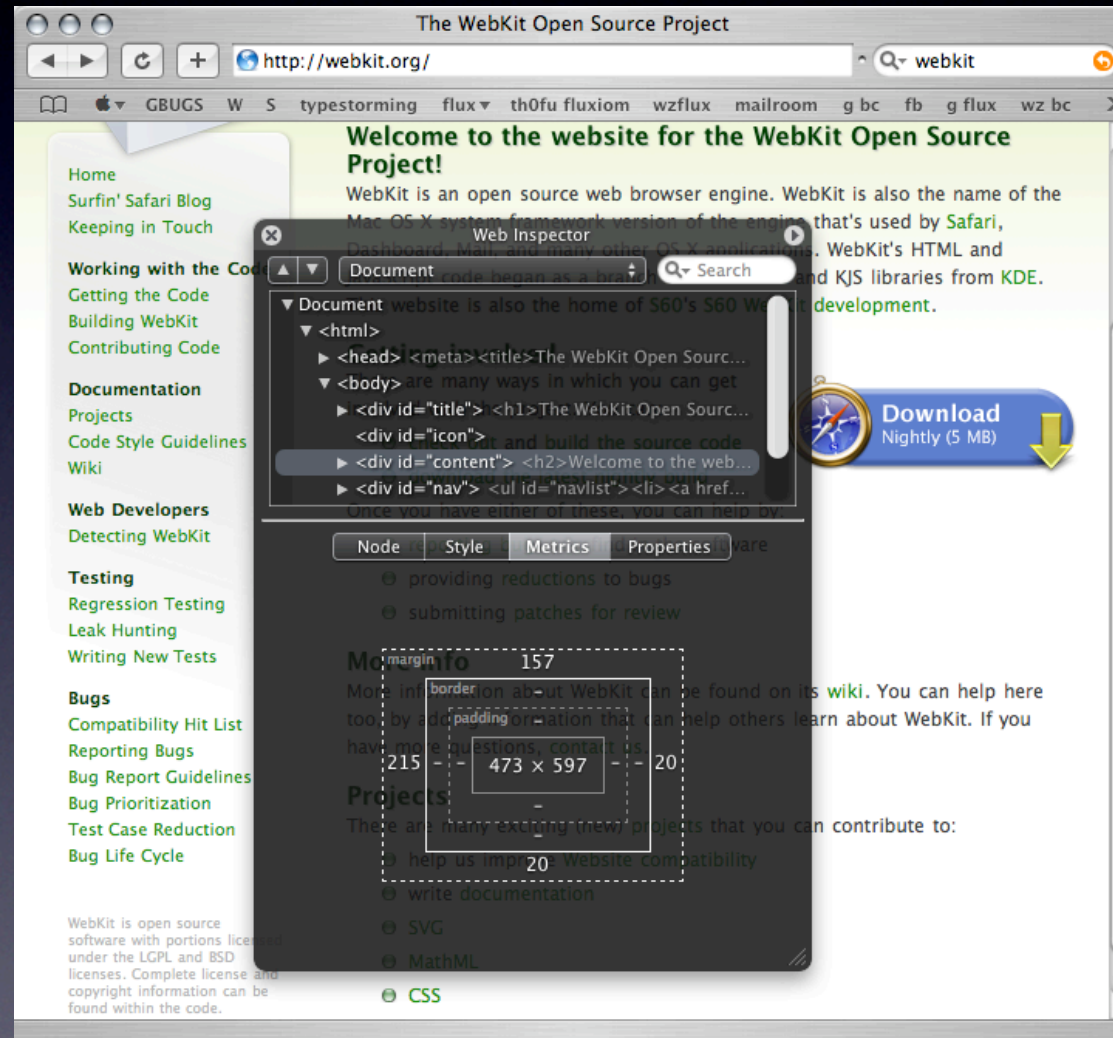
>>> Fertig

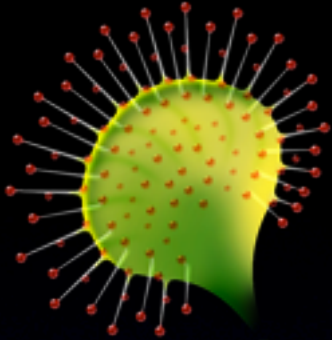
Venkman JavaScript Debugger





Safari Web Inspector





Safari Debugging with Drosera

„script.aculo.us – web 2.0 javascript“ wird geladen

http://script.aculo.us/ RSS Google

GBUGS W S typestorming flux th0fu fluxiom wzflux mailroom g bc fb g flux wz bc wolke wollzelle edgedoc del.icio.us

script.aculo.us
it's all about the user interface, baby!

web.reload!

script.aculo.us provides you with easy-to-use, cross-browser user interface JavaScript libraries to make your web sites and web applications fun.

What's inside? [animation framework](#), [drag and drop](#), [Ajax controls](#), [DOM utilities](#), and [unit testing](#).

It's an add-on to the fantastic [Prototype](#) framework.

Subscribe to RSS updates
Stay informed on new versions and important bug fixes

WebKit – Debugger

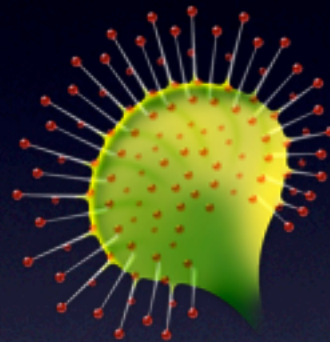
Continue Pause Step Into Step Out Step Over Console

#	Function	Variable	Value
178			
179			
180			
181			
182			
183			
184			
185			
186			
187			
188			
189			
190			
191			
192			
193			

```
178 }
179
180 $('header').setStyle({top:'170px',left:'-700px'});
181 $('logo').setStyle({display:'block'});
182
183 new Effect.Move('header',{ x: 730, y: -170 });
184
185 function bubble(id,x,y){
186   $(id+'-bubble').setStyle({left:x+'px',top:y+'px'});
187   new Effect.Scale(id+'-bubble',100, Object.extend({
188     beforeStart:function(effect){
189       $(effect.element).style.display = 'block';
190       $(effect.element).setOpacity(0);
191       $$('#'+id+'-bubble p').each(function(p){p.hide()});
192     },
193     afterUpdate:function(effect){f
```




+



<http://webkit.org/>

Microsoft Script Debugger for IE



<http://blogs.msdn.com/ie/archive/2004/10/26/247912.aspx>

Test with all
browsers
you want to
support




```
config.action_view.debug_rjs = true
```



http://localhost:3000

RJS error:

TypeError - Undefined value

OK



http://localhost:3000

`$("boo").visualEffect("highlight");`

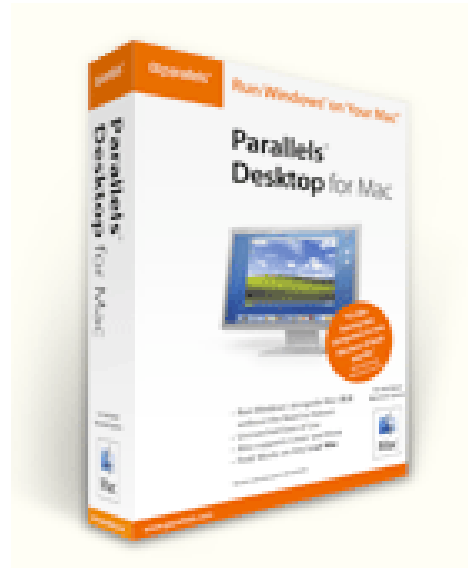
OK

get a setup that
doesn't suck





+



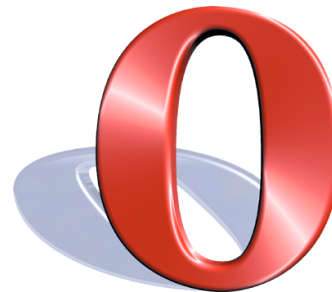
+



(+  ubuntu)



one machine to rule them all



help and ideas welcome

#prototype (freenode)
and

Rails Spinoffs Google Group