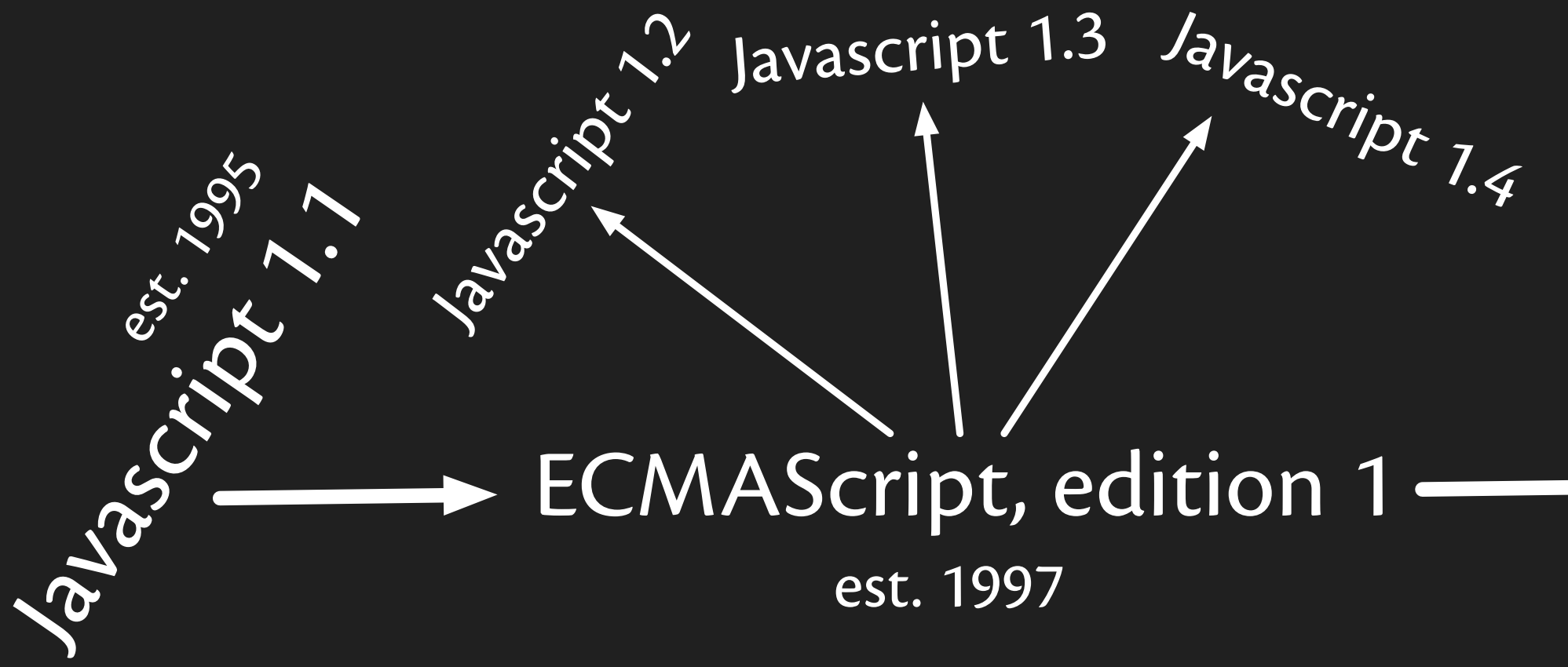
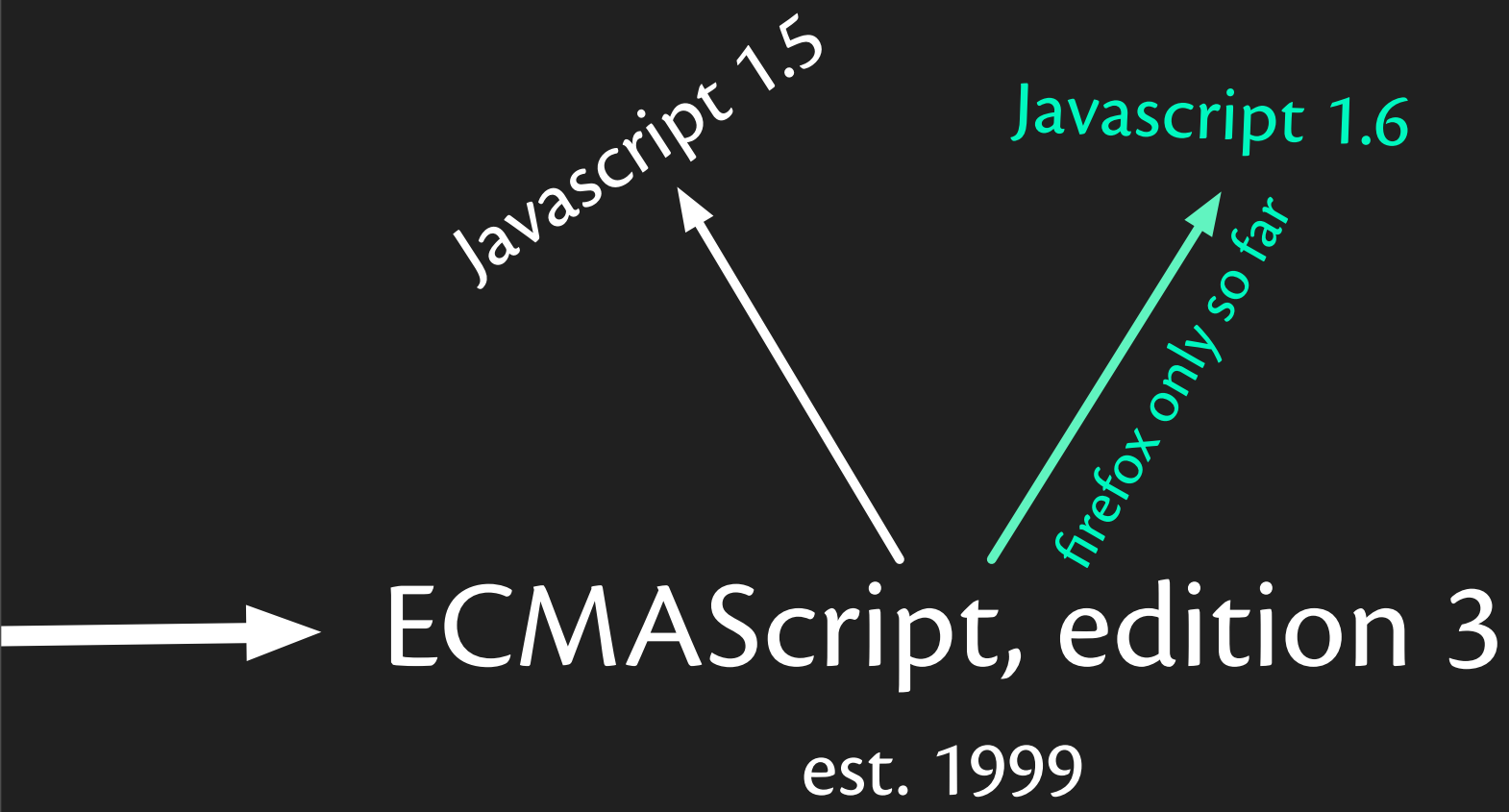
javascript BOOTCAMP

by amy hoy, www.slash7.com

Javascript is a *real* ~~boy~~ language


est. 1995
Javascript 1.1
Javascript 1.2
Javascript 1.3
Javascript 1.4
ECMAScript, edition 1
est. 1997


Javascript 1.5
Javascript 1.6
firefox only so far
ECMAScript, edition 3
est. 1999

Follow Along

www.squarefree.com/shell/shell.html

Syntax & Style

```
//// Semi-colons are not required
var foo = getBar()
// --> [no error]
var foo = getBar() fooBar()
// ! SYNTAX ERROR !

//// CamelCase is the norm
if(fooBar == bazBat) {}

//// Object attributes & methods attached with '.'
someObject.someFunction()
```

whirlwind tour

look ma, no primitives!

```
var string = "Hello, I'm a string!"
// --> "Hello, I'm a string!"

string.length
// --> 20

"Hello, I'm a string!".length
// --> 20
```

data types are fun!

* Number
* String
* Boolean
* null
* undefined
* RegExp

creating numbers

```
var number = 5
// --> 5

var anotherNumber = new Number(5)
// --> 5

var pi = 3.14
// --> 3.14
```

yes, they're all numbers

```
typeof number
// --> "number"

typeof anotherNumber
// --> "number"

typeof pi
// --> "number"
```

creating strings

```
var string = "Hello, I'm a string!"
// --> "Hello, I'm a string!"

var aRealString = new String("I'm a REAL string!")
// --> [I,',m, ,a, ,R,E,A,L, ,s,t,r,i,n,g,!]
aRealString
// --> "I'm a REAL string!"

var anotherString = "Hello, I'm \
a string with \
linebreaks!"
// --> "Hello, I'm
// a string with
// linebreaks!"
```

creating regexps

```
var string = "string1 string2 string3"
string.replace(/string1/, 'String Spectacular')
// --> "String Spectacular string2 string3"

var string = "string1 string2 string3"
string.replace(new RegExp("string1"), 'String Spectacular')
// --> "String Spectacular string2 string3"
```

booleans - dead simple

```
var booleanVar = false
// --> false

var anotherBooleanVar = true
// --> true

if(booleanVar) alert("Test!")
// --> *nothing*

if(!booleanVar) alert("Test!")
// --> [alert] Test!

var anotherBoolean = iReturnABool()
```

null and undefined are special

```
//// null is case-sensitive
booleanVar = null
// --> null

//// undefined (also case-sensitive)
booleanVar == undefined
// --> true [because it was set to null]

booleanVar === undefined
// --> false
```

doin' stuff with basic types

```
3 * 5 + 9
// --> 24

3 * (5 + 9)
// --> 42

"string1 string2 string3".length
// --> 23

var start = "string1 string2 string3".indexOf('string2')
// --> 8

"string1 string2 string3".substr(start)
// --> "string2 string3"
```

creating arrays

```
var emptyArray = []
// --> []

emptyArray.length
// --> 0 [the number, not false]

var anArray = ['foo','bar','baz']
// --> ['foo','bar','baz']
```

```
var array = [] // --> []
```

this deserves repeating

creating simple functions

```
function noArgument() {
    // do stuff
    alert("I can't argue; I have no arguments")
}

function simpleFunction(arg1,arg2,arg3) {
    // do stuff here
    return arg1 + arg2 + arg3
}

noArgument()
// --> [alert] I can't argue; I have no arguments

simpleFunction(1,2,3)
// --> 6
```

creating simple objects

```
var newObject = {}
// --> {} (empty object!)

newObject.foo = "bar"
// --> "bar"

newObject.foo
// --> "bar"

var anotherObject = {foo:"bar", baz:"bat"}
// --> {foo:"bar", baz:"bat"}

anotherObject.baz
// --> "bat"

anotherObject.baz = "zort"
// --> "zort"
```

```
var newObject = {} // --> {}
```

this deserves repeating

if... or else!

```
if(something) doStuff()

if(something)
    alert("True!")
else
    alert("Not true!")

if(something == foobar) {
    alert("equals foobar!")
} else if(something == bazbat) {
    alert("equals bazbat!")
} else {
    alert("equals neither!")
}
```

gonna have to make a switch()

```
switch(something) {
    case "foobar":
        alert("Oh no! It's a foobar!")
        break
    case "barfoo":
        alert("Barfoo!")
        break
    case "fallthru":
        alert("Falling through...")
    case "fellthru":
        alert("fallen through.")
        break
    default:
        alert("Case not found... here's a default")
}
```

while()

```
var i = 0
while(i < 3) {
    alert(i)
    i++
}
// --> [alert] 0, [alert] 1, [alert] 2
```

do..while() while() while()

```
var i = 0
do {
    alert(i)
    i++
} while(i < 3)
// --> [alert] 0, [alert] 1, [alert] 2
```

for(), our old friend

```
for(var i = 0; i < 3; i++) {
    alert(i)
}
// --> [alert] 0, [alert] 1, [alert] 2
```

for..in(), a new trick!

```
var theObject = {foo:"bar", baz:"bat", narf:"zoit"}
// --> [Object object]

for(attribute in theObject) {
    alert(attribute +" = "+ theObject[attribute])
}
// --> [alert] foo = bar, [alert] baz = bat
//     [alert] narf = zoit
```

strings

strings act like arrays

```
var string = "Hello, I'm a string!"
// --> "Hello, I'm a string!"

string[0]
// --> H
string[3]
// --> l

for(var i = 0; i < string.length; i++) {
    alert(string[i])
}
// --> [alert] H, [alert] e, [alert] l, etc.
```

Concatenation & auto type conversion

```
"string1 " + " string2"
// --> "string1  string2"

"string " + 5
// --> "string 5"

5 + "5"
// --> "55"

"string".length
// --> 6
```

breakin' up is not hard to do

```
"string1 string2 string3".indexOf('string2')
// --> 8

"string1 string2 string3".substr(8)
// --> "string2 string3"

var string = "string1 string2 string3"
string.substr(string.indexOf('string2'))
// --> "string2 string3"

//// string was not changed:
string.split(' ')
// --> ["string1", "string2", "string3"]
```

the great escape. and unescape.

```
"<h3>Here's a headline!</h3>".escape()
// ! TypeError !
var escaped = escape("<h3>Here's a headline!</h3>")
// --> "%3Ch3%3EHere%27s%20a%20headline%21%3C/h3%3E"
var unescaped = unescape(escaped)
// --> "<h3>Here's a headline!</h3>"

var URL = 'http://mysite.com/?stuff="Foo bar!"&bar="stuff"'
var encodedURL = encodeURI(url)
// --> http://mysite.com/?stuff=%22Foo%20bar!%22&bar=%22stuff%22
var decodedURL = decodeURI(encodedURL)
// --> http://mysite.com/?stuff="Foo bar!"&bar="stuff"
```

RegExps

a bit more on regexps

```
var simpleRegExp = /\/foo/g
var anotherRegExp = new RegExp("/foo","g")

"/foobar /foof /foobaz".match(simpleRegExp)
// --> ["/foo", "/foo", "/foo"]

"/foobar /foof /foobaz".match(anotherRegExp)
// --> ["/foo", "/foo", "/foo"]

"/foobar /foof /foobaz".match(/zort/)
// --> null

simpleRegExp.test("/foobar /foof /foobaz")
// --> true
```

functions

closures & callbacks

```
function myFunction(arg1,arg2,arg3) {
    // use argument values
    alert("I have an argument! " + arg1)
    // use an object argument
    alert(arg2.bar)
    // call an argument as a function
    arg3()
}

myFunction("foo", {bar:"baz"}, function(){ alert("Victory!")})
// outputs...
//      [alert] I have an argument! foo
//      [alert] baz
//      [alert] Victory!
```

function handles

```
//// Creating a function handle
var myOtherFunction = function() {just
    alert("Victory!")
}

myFunction("foo", {bar:"baz"}, myOtherFunction)
```

the arguments object

```
function myFunction() {
    alert("I have an argument! " + arguments[0])
    alert(arguments[1].bar)
    arguments[2]()
}

myFunction("foo", {bar:"baz"}, myOtherFunction)
// outputs...
//      [alert] I have an argument! foo
//      [alert] baz
//      [alert] Victory!
```

objects

JavaScript Object Notation (JSON)

```
var foo = {
    bar:"baz",
    zort:"narf",
    aNumber:5,
    anArray:['banana','plaintain','ugli fruit'],
    doStuff:function() {
        alert("I'm doing stuff!")
    }
}
// --> [Object object]
```

using a JSON object

```
foo.zort
// --> "narf"

for(i = 0; i < foo.anArray.length; i++) {
    alert(foo.anArray[i])
}
// --> [alert] banana, [alert] plaintain,
//     [alert] ugli fruit

foo.doStuff()
// --> [alert] I'm doing stuff!
```

turning strings into code (more JSON!)

```
var data = "{banana:'yummy', plaintain:'icky'}"

function handleAjaxResult(ajaxResult) {
    eval("var resultData = "+ajaxResult)
    alert(resultData.banana)
}

handleAjaxResult(data)
// --> [alert] yummy
```

don't forget your commas

```
var foo = {
    bar:"baz",
    zort:"narf",
    aNumber:5,
    anArray:['banana','plaintain','ugli fruit']
    doStuff:function() {
        alert("I'm doing stuff!")
    }
}
// --> ! SyntaxError ! Line 6: missing } after property list
```

creating an object constructor

```
function anyFunction() {}

var object = new anyFunction()
// --> [Object object]
```

creating a better object constructor

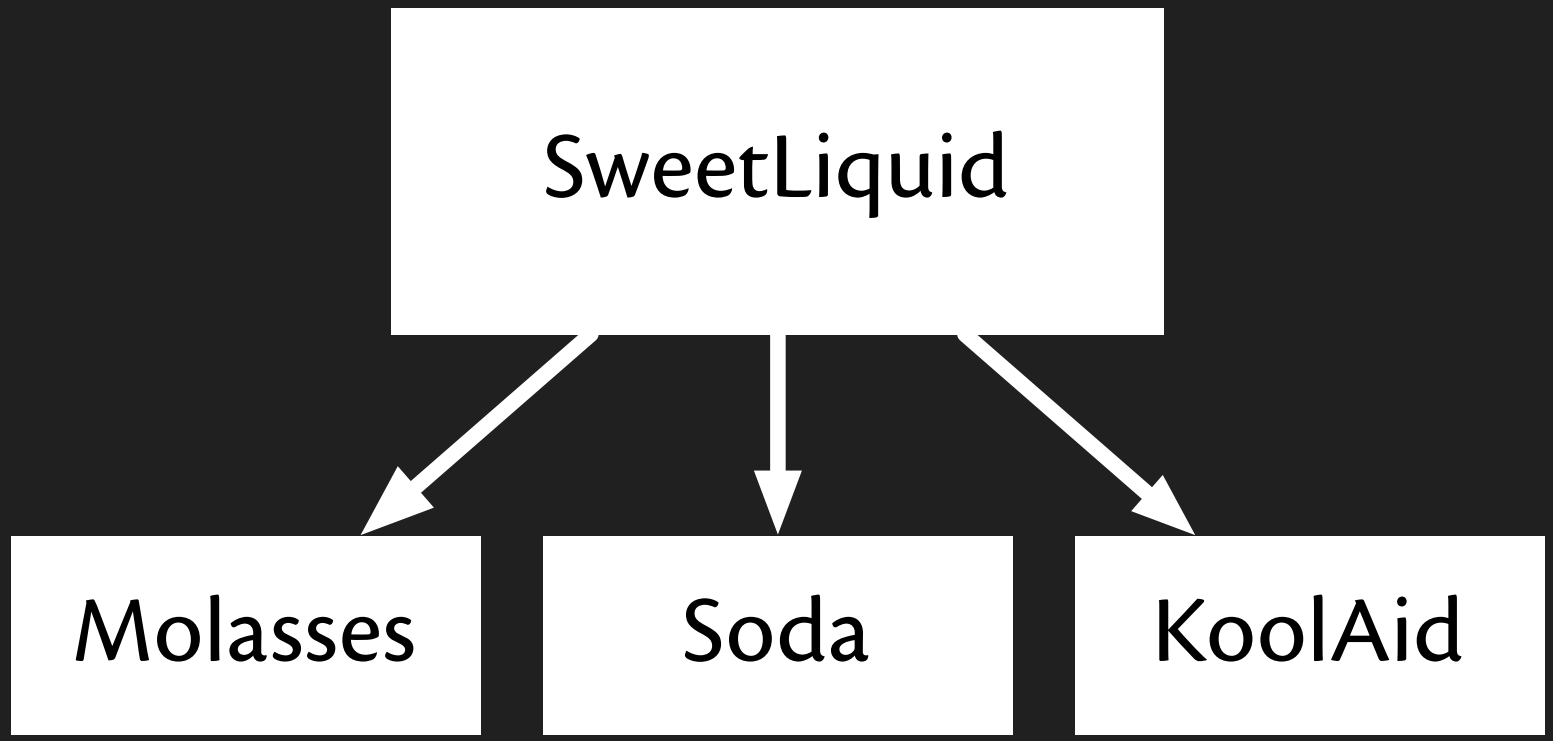
```
function Foo() {
    // use this keyword inside function constructors
    this.bar = "baz"
    // run arbitrary code on construction
    alert("I'm a new Foo!")
}

var foo = new Foo()
// --> [Object object] & [alert] I'm a new Foo!

foo.bar
// --> "baz"
```

Javascript has a prototype-based inheritance model

using Prototype to extend objects

```
function Foo() {
    this.bar = "baz"
    alert("I'm a new Foo!")
}

var fooInstance = new Foo() // --> [alert] I'm a new Foo!
fooInstance.bar // --> "baz"

Foo.prototype.bar = "bat"
Foo.prototype.fuzz = "boo"
Foo.prototype.danger = function() { alert("Alert! Alert!" ) }

fooInstance.danger() // --> [alert] Alert! Alert!
fooInstance.bar //--> "baz"
fooInstance.fuzz // --> "boo"
```

creating an object hierarchy



creating the parent object

```
function SweetLiquid() {
    this.volume = 500
    this.unit = 'ml'
    this.drink = function() { this.volume-- }
    this.sweetnessRating = .5
}
```

creating the children

```
function Soda() {}
function Molasses() {
    this.viscosity = "low"
}

function KoolAid() {
    alert("OHH YEAH!")
    this.mascot = "awesome"
    this.unit = 'quart'
    this.volume = '4'
}
```

creating the relationship

```
Soda.prototype = new SweetLiquid
Molasses.prototype = new SweetLiquid
KoolAid.prototype = new SweetLiquid
```

mucking about with your new objects

```
var cola = new Soda // --> [Object object]
cola.volume // --> 500
cola.unit // --> "ml"

var strawberryKoolAid = new KoolAid
// --> [Object object] ... [alert] "OHH YEAH!"
strawberryKoolAid.mascot // --> "awesome"
strawberryKoolAid.unit // --> "quart"
strawberryKoolAid.volume // --> 4
```

further extending objects

```
KoolAid.prototype.drink = function(amount) {
    this.volume = this.volume - amount
    return this.volume
}

strawberryKoolAid.drink(1) // --> 3
```

extending specific instances

```
strawberryKoolAid.spill = function(amount) {
    this.volume = this.volume - (amount * 2)
    return this.volume
}

grapeKoolAid = new KoolAid
// --> [Object object] ... [alert] "OHH YEAH!"

strawberryKoolAid.spill(1) // --> 1
grapeKoolAid.spill(1)
// ! TypeError ! grapeKoolAid.spill is not a function
```

exceptions

try..catch

```
grapeKoolAid.spill(1)
// ! TypeError ! grapeKoolAid.spill is not a function

try {
    grapeKoolAid.spill(1)
} catch (e) {
    alert("Oops! An error of type "+e.name + " occurred. \
    The message was: " +e.message)
}
// --> [alert] Oops! An error of type TypeError occurred.
//     The message was: grapeKoolAid.spill is not a function
```

fancier try..catch

```
try {
    grapeKoolAid.spill(1)
} catch (e if e.name == "TypeError") {
    alert("Oops! You seemed to have called the wrong \
    kind of thing")
} catch (e) {
    alert("Another kind of error has occurred")
}
// --> [alert] Oops! You seemed to have called the wrong
//     kind of thing
```

throwing custom exceptions

```
grapeKoolAid.spill = function(amount) {
    if(amount) {
        throw "You can't spill grape KoolAid! \
        That's ridiculous!"
    }
}
grapeKoolAid.spill(2)
// ! CantSpillError ! You can't spill grape KoolAid!
//  That's ridiculous!
```

throwing custom exceptions

```
grapeKoolAid.spill = function(amount) {
    if(amount) {
        throw "You can't spill grape KoolAid! \
        That's ridiculous!"
    }
}
grapeKoolAid.spill(2)
// ! CantSpillError ! You can't spill grape KoolAid!
//  That's ridiculous!
```

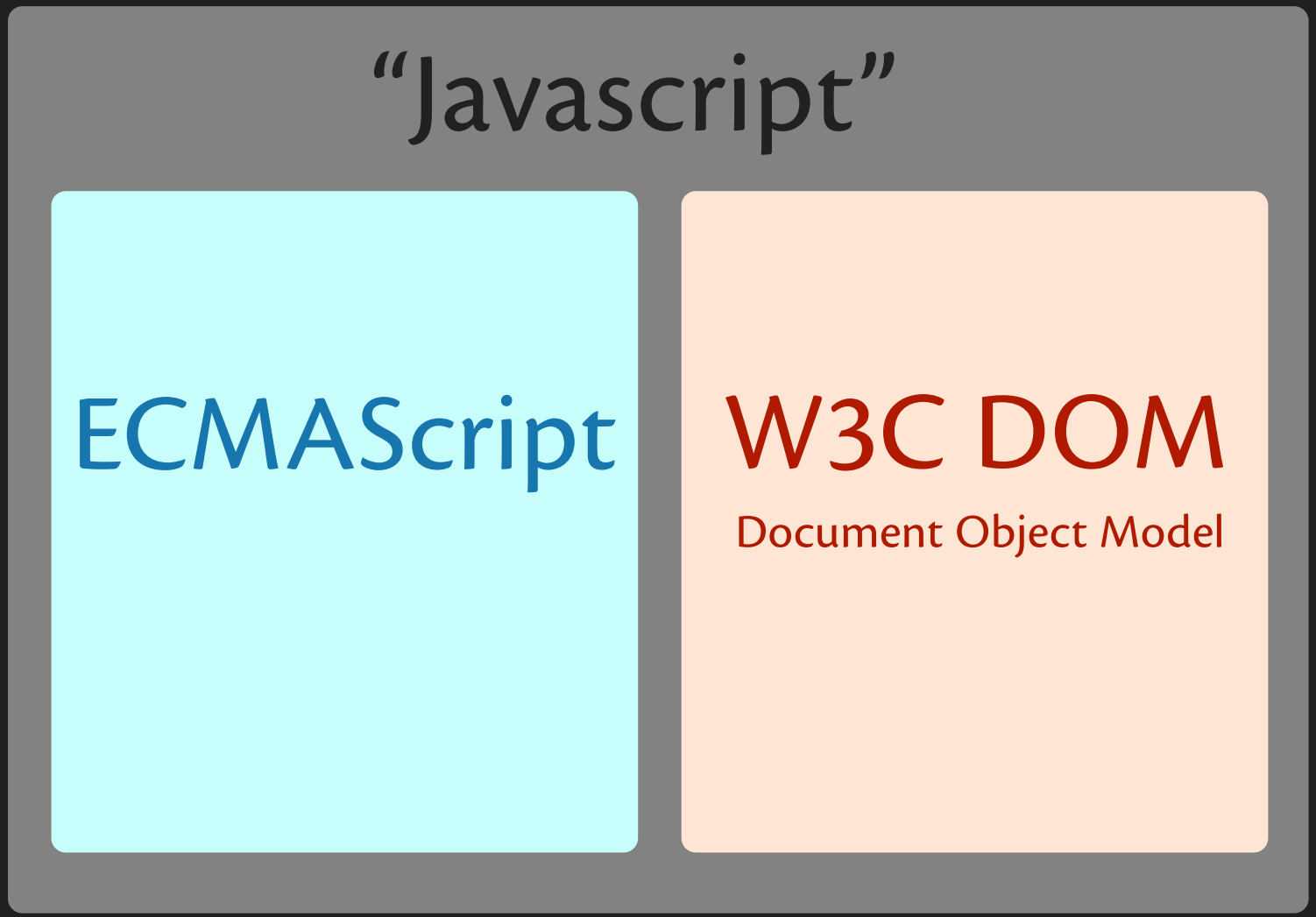
throwing custom exception objects

```
function CantSpillError(beverage) {
    this.name = "CantSpillError"
    this.message = "You can't spill "+beverage+"! \
    That's ridiculous!"
}

grapeKoolAid.spill = function(amount) {
    if(amount) {
        throw new CantSpillError("grape KoolAid")
    }
}
```

catching custom exceptions

```
try {
    grapeKoolAid.spill(1)
} catch (e) {
    alert("Oops! An error of type "+e.name + " occurred. \
    The message was: " +e.message)
}
// --> [alert] Oops! An error of type CantSpillError occurred.
//     The message was: You can't spill grape KoolAid!
//     That's ridiculous!
```

the browser



the split-personality language

Javascript the language is mostly compatible

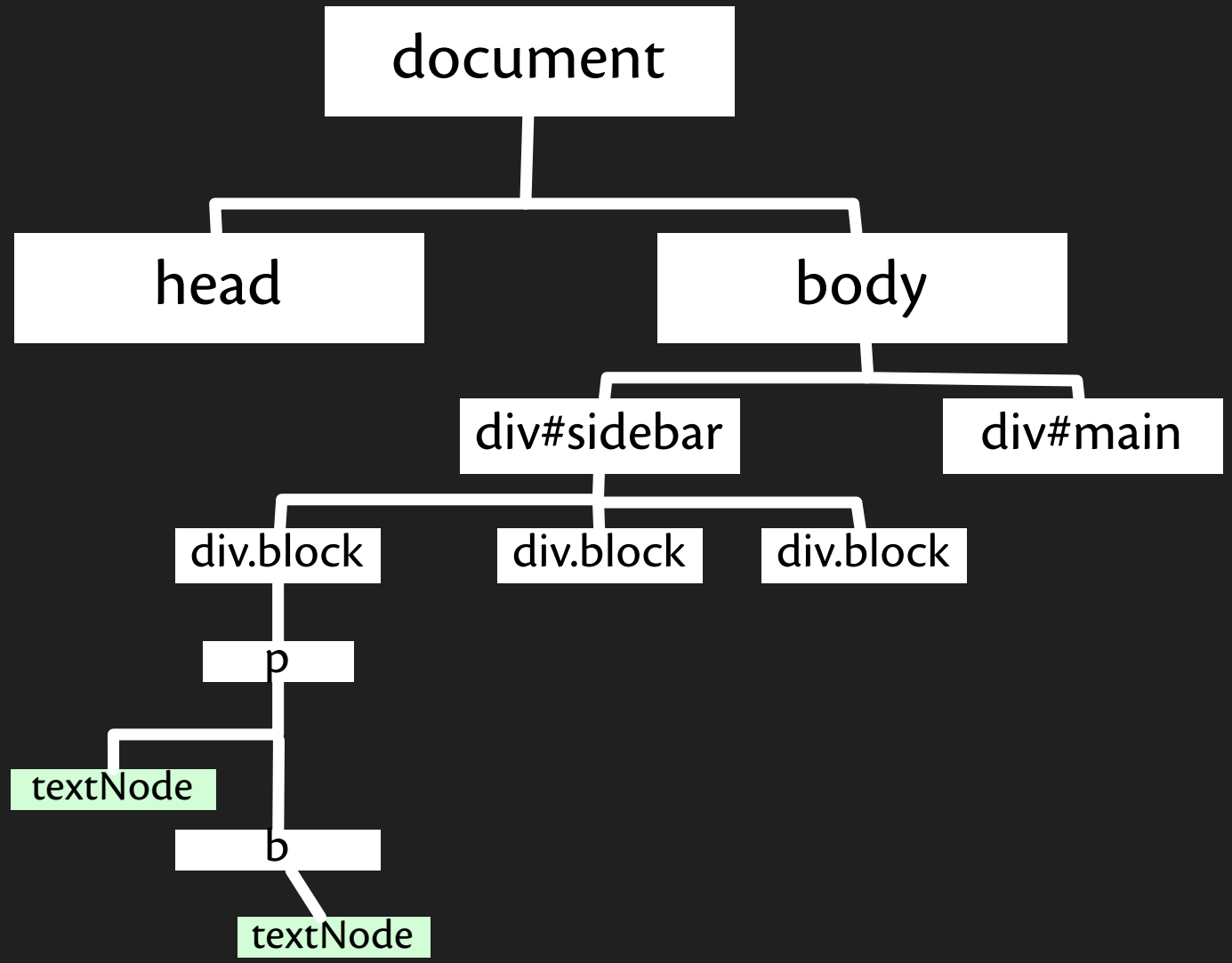
the DOM is the problem *area*

IE is *the problem*, but not as bad as you'd think

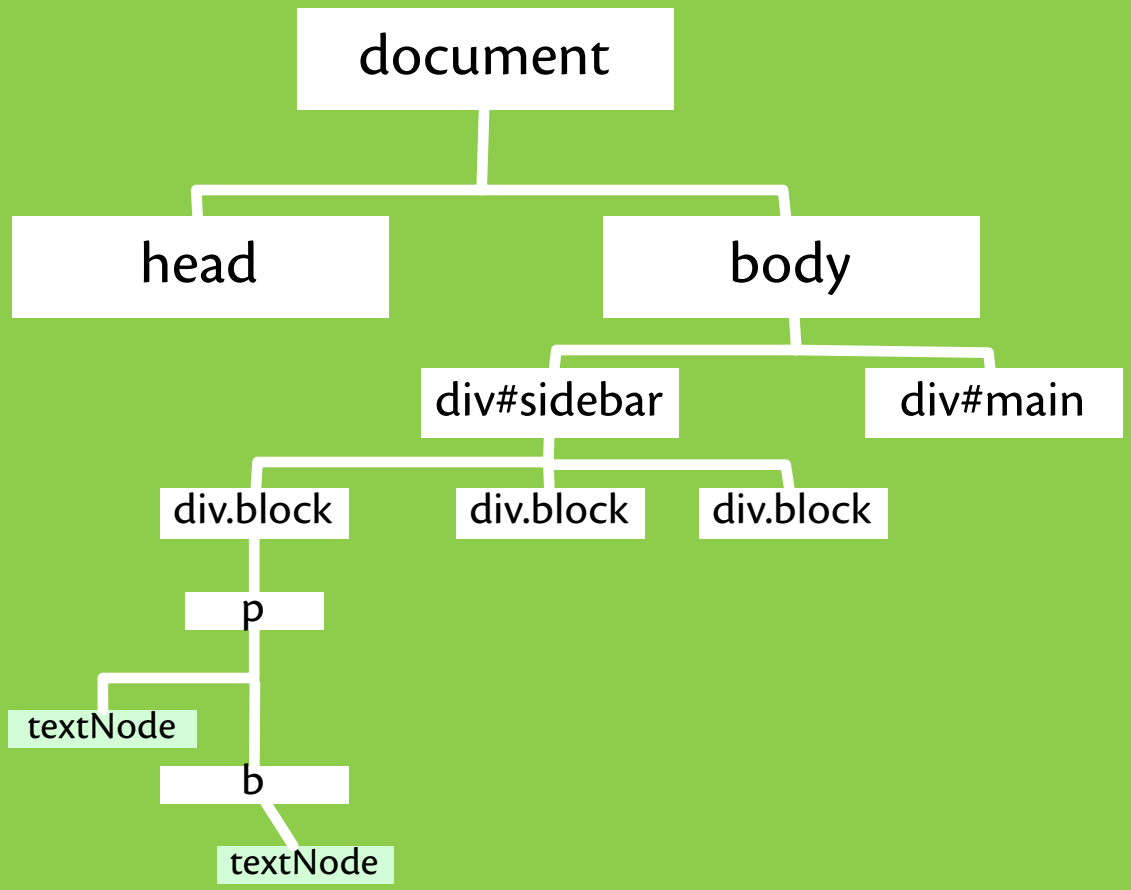
Don’t hate the language, hate the runtime environment

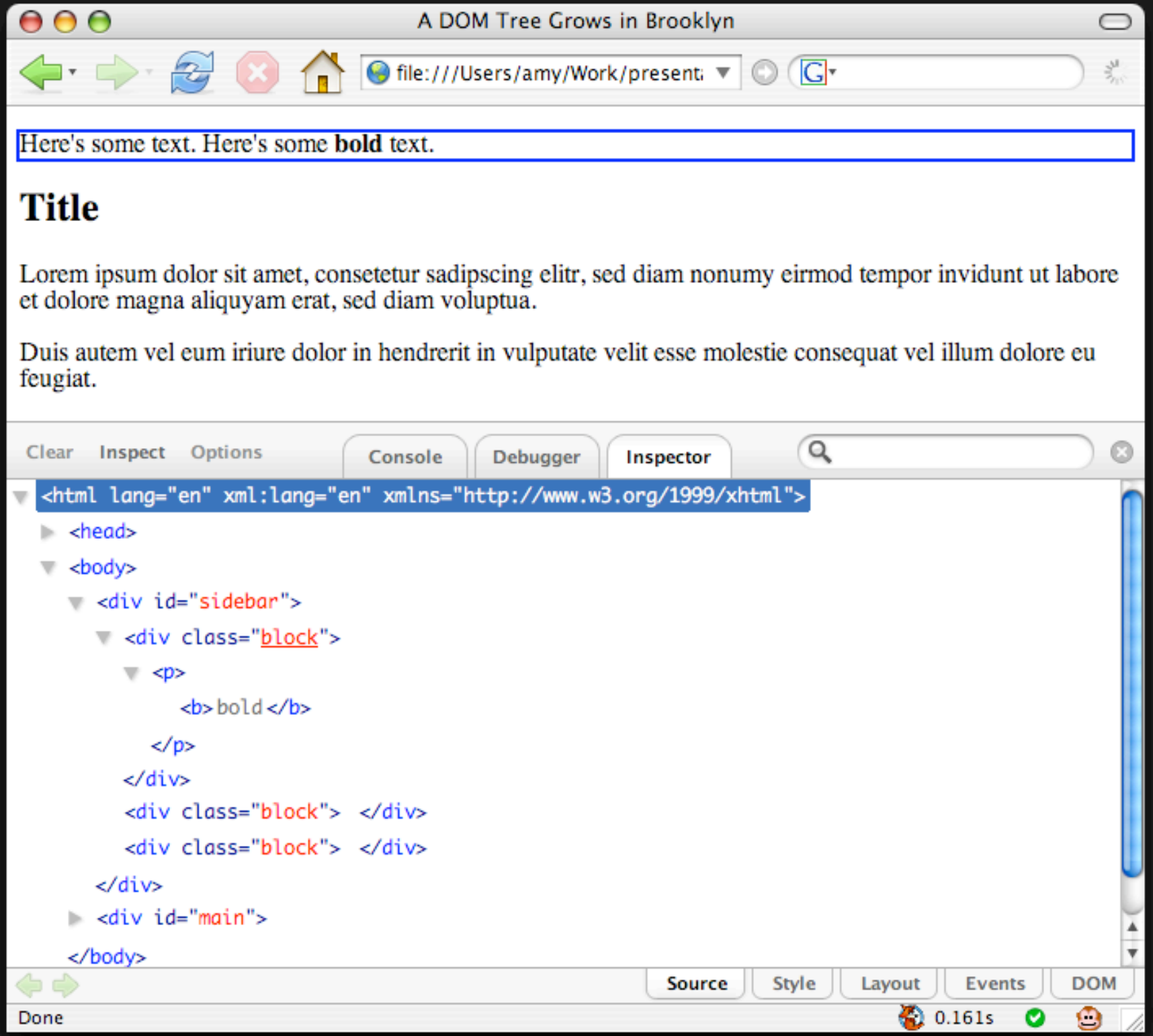
the DOM

```
<!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>
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
        <title>A DOM Tree Grows in Brooklyn</title>
    </head>
    <body>
        <div id="sidebar">
            <div class="block">
                <p>Here's some text. Here's some <b>bold</b> text.</p>
            </div>
            <div class="block">
            </div>
            <div class="block">
            </div>
        </div>
        <div id="main">
            <div class="post">
                <h2>Title</h2>
                <p>
                    Lorem ipsum dolor sit amet, consetetur sadipscing elitr,
                    sed diam nonumy eirmod tempor invidunt ut labore et dolore
                    magna aliquyam erat, sed diam voluptua.
                </p>

                <p>
                    Duis autem vel eum iriure dolor in hendrerit in vulputate
                    velit esse molestie consequat vel illum dolore eu feugiat.
                </p>
            </div>
        </div>
    </body>
</html>
```


document
head
body
div#sidebar
div#main
div.block
div.block
div.block
p
textNode
b
textNode

window


document
head
body
div#sidebar
div#main
div.block
div.block
div.block
p
textNode
b
textNode


A DOM Tree Grows in Brooklyn
file:///Users/amy/Work/present
Here's some text. Here's some bold text.
Title
Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua.
Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat vel illum dolore eu feugiat.
Clear
Inspect
Options
Console
Debugger
Inspector
<html lang="en" xml:lang="en" xmlns="http://www.w3.org/1999/xhtml">
<head>
<body>

bold

</body>
Source
Style
Layout
Events
DOM
Done
0.161s

iterating through the DOM tree

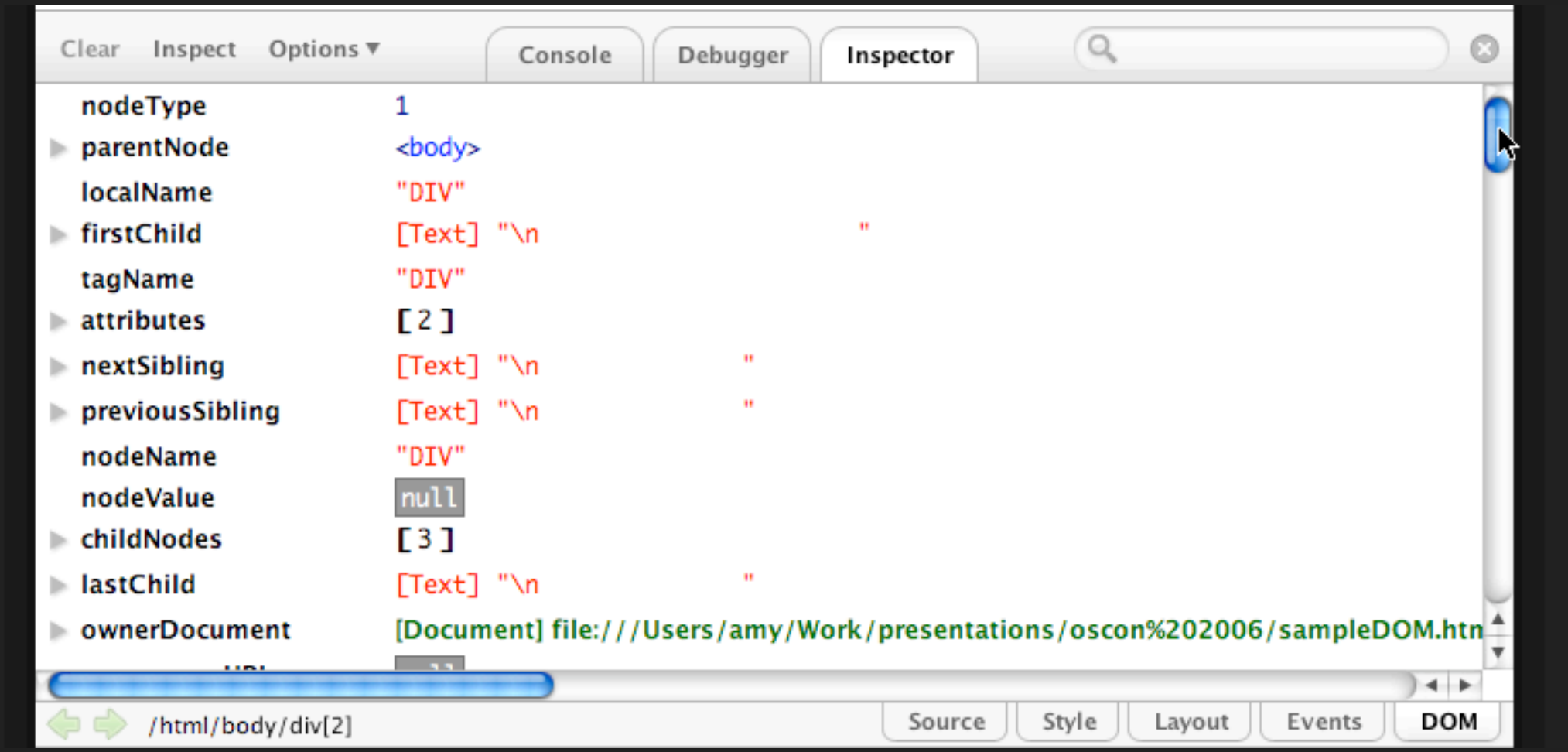
```
var children = document.childNodes;
for(var i=0; i < children.length; i++) {
    alert(children[i])
}
// --> [alert] [Object DocumentType]
// --> [alert] [Object HTMLHtmlElement]

var children = document.body.childNodes;
for(var i=0; i < children.length; i++) {
    alert(children[i])
}
// --> [alert] [Object Text]
// --> [alert] [Object HTMLDivElement]
// --> [alert] [Object Text]
// etc
```

DOM element attributes

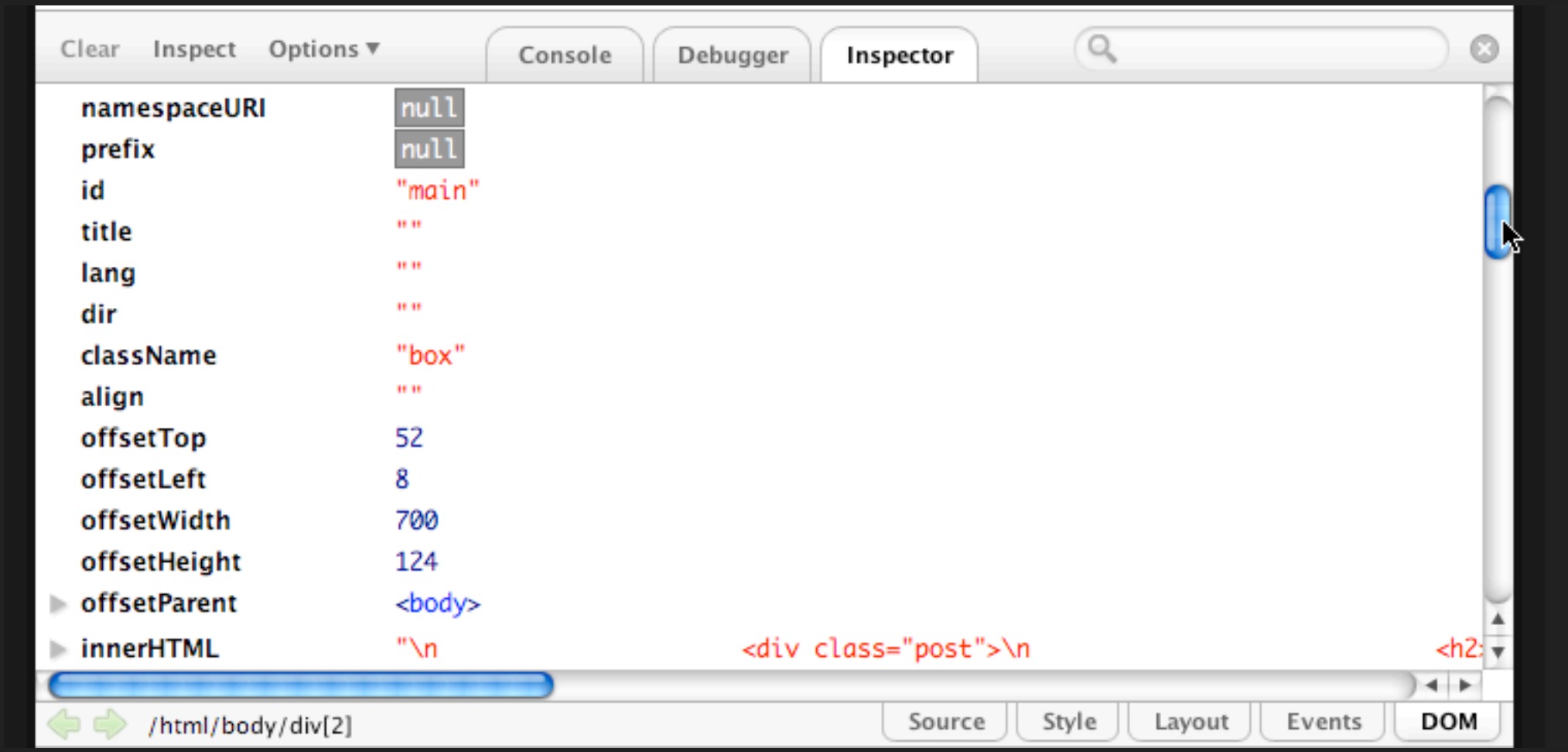
```
<div id="main" class="box">
    <div class="post">
        <h2>Title</h2>
        <p>
            Lorem ipsum dolor sit amet, consetetur sadipscing elitr,
            sed diam nonumy eirmod tempor invidunt ut labore et dolore
            magna aliquyam erat, sed diam voluptua.
        </p>
    </div>
</div>
```

DOM element attributes

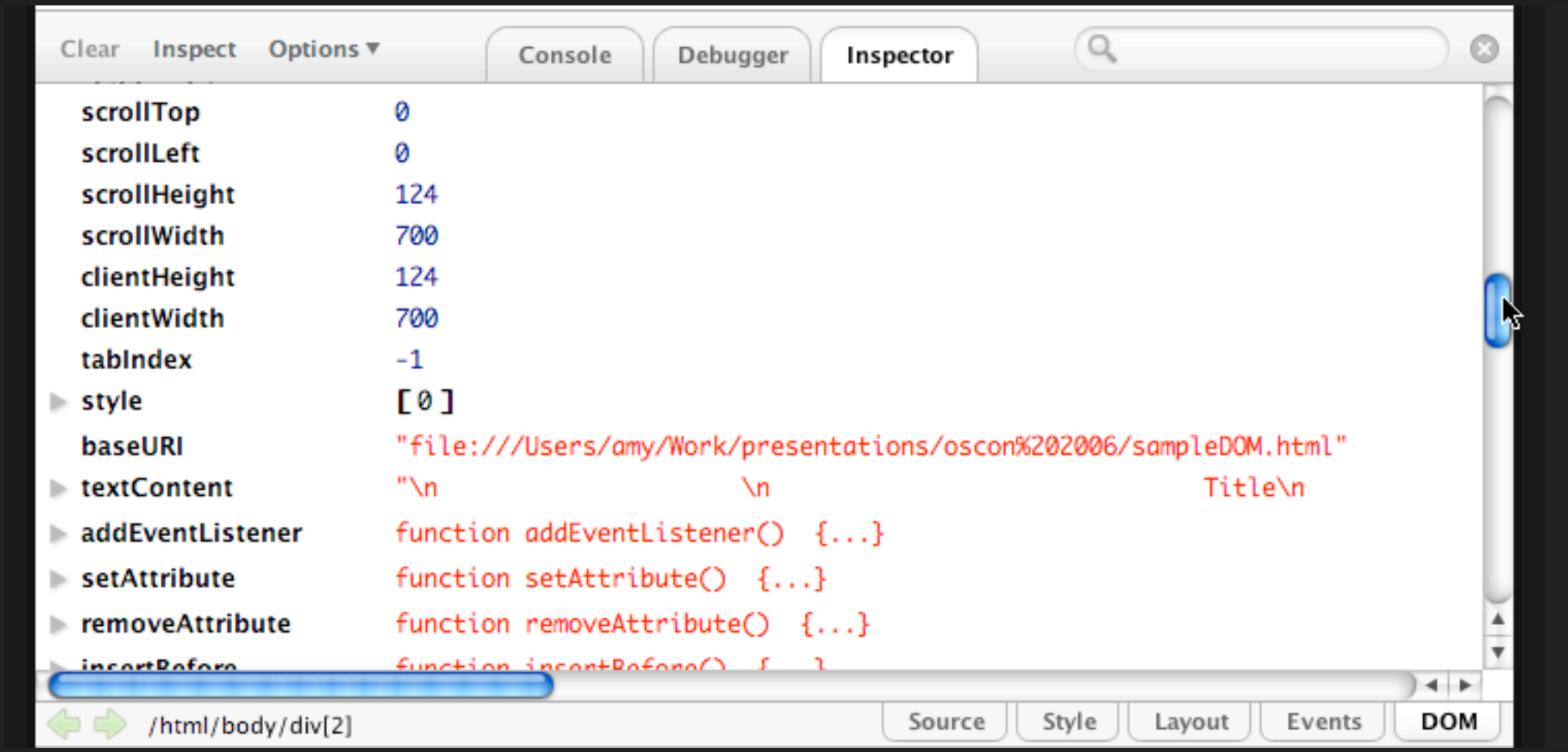
DOM element attributes



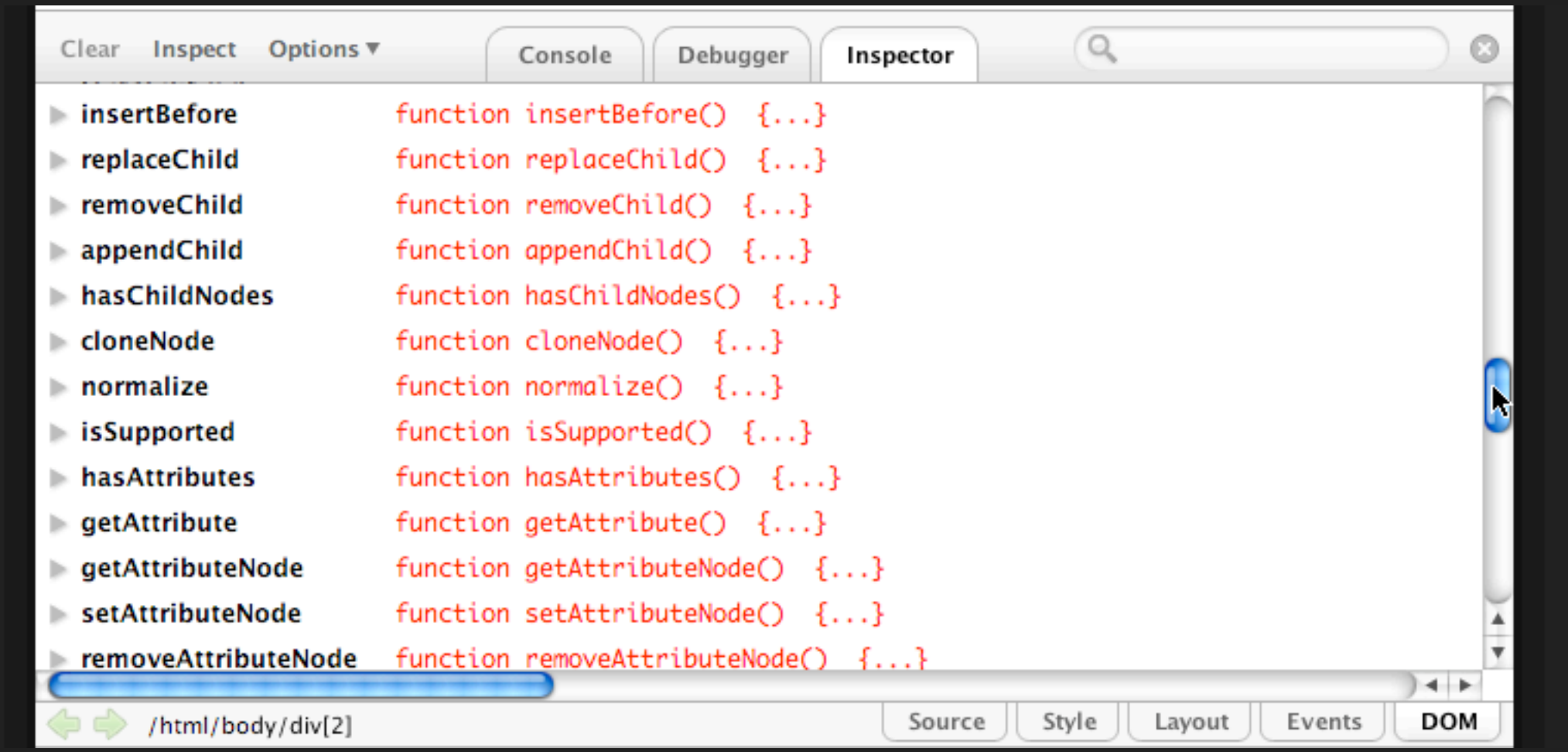
DOM element attributes



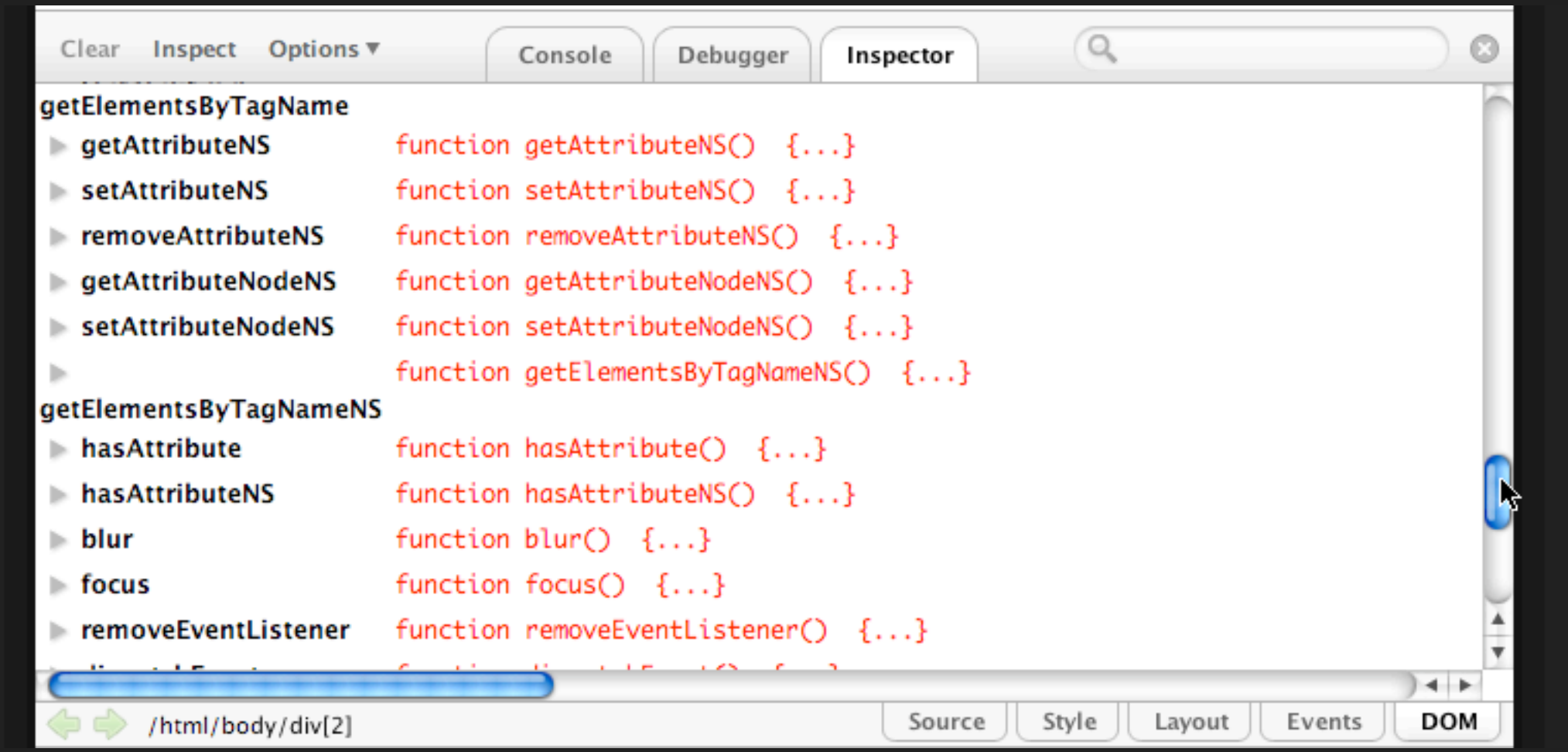
DOM element attributes & methods



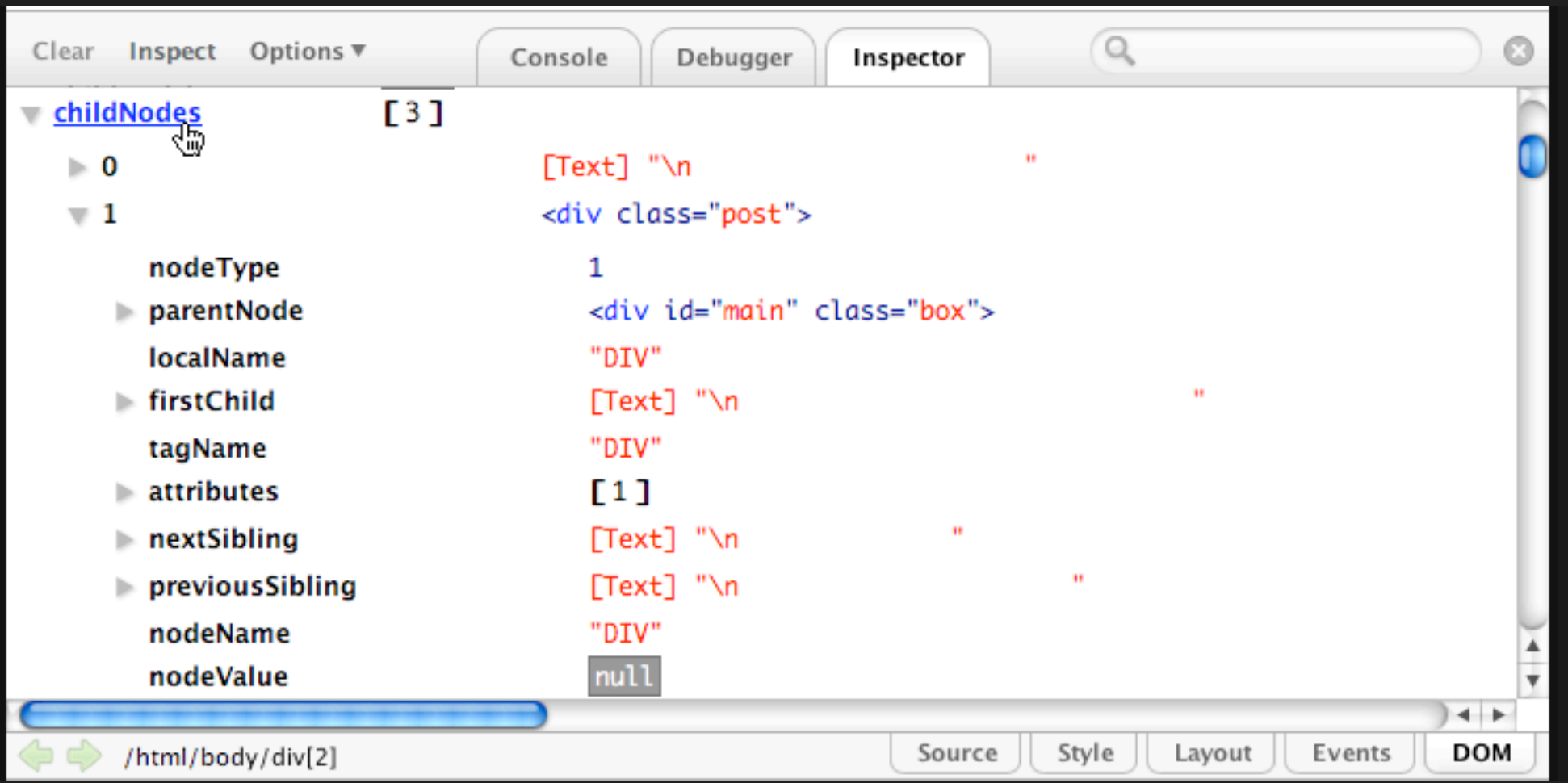
DOM element attributes & methods



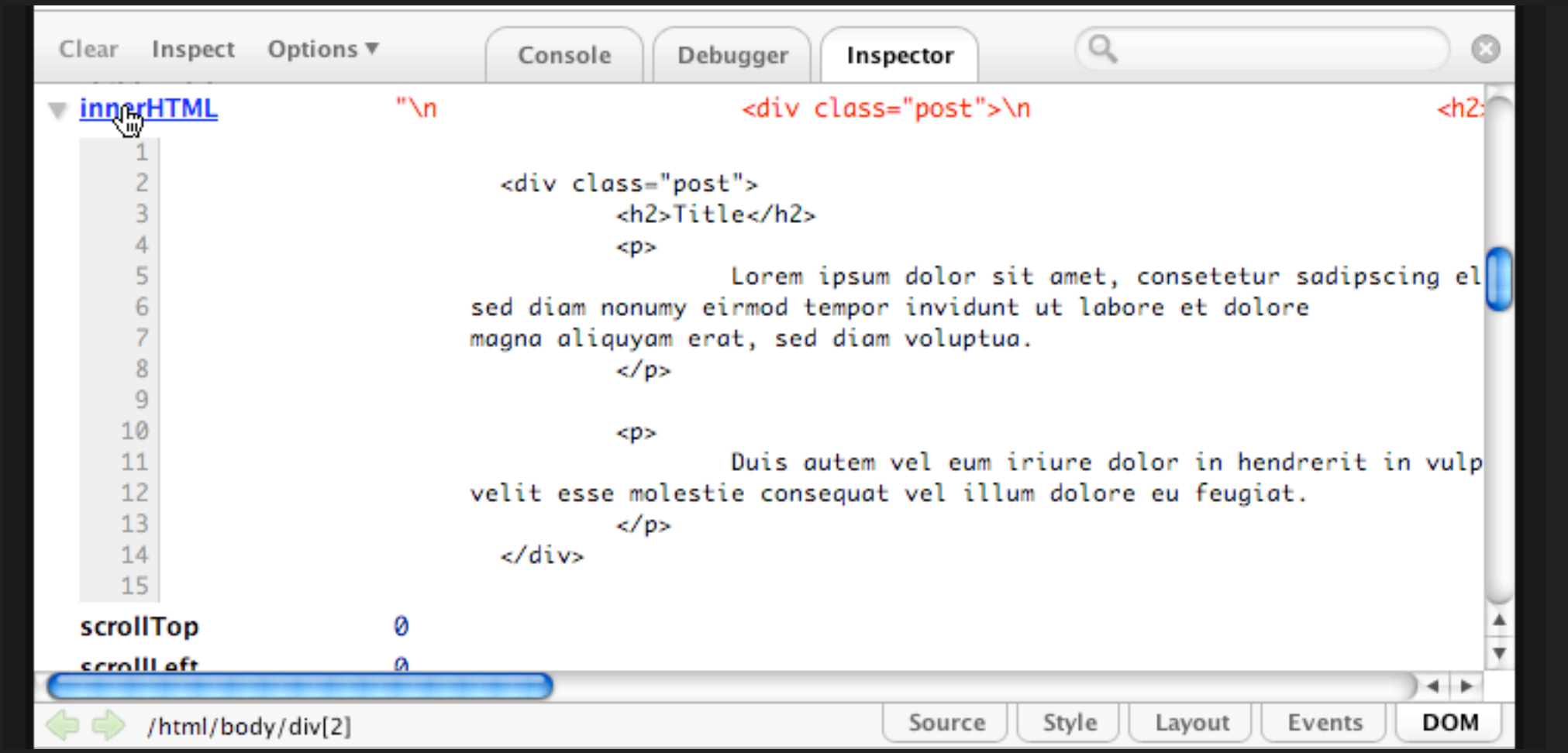
DOM element attributes & methods



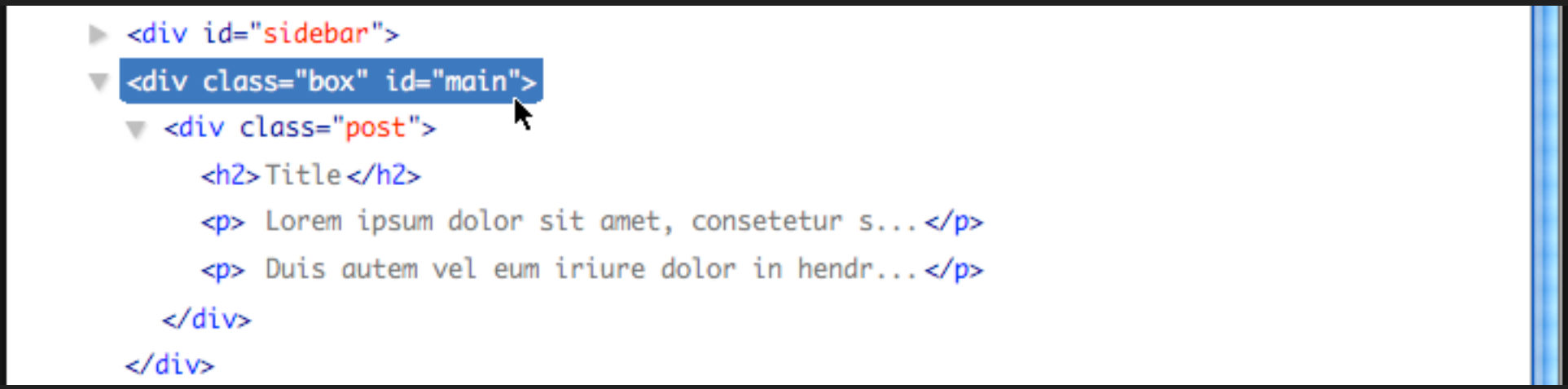
node.childNodes



node.innerHTML



finding DOM elements by id



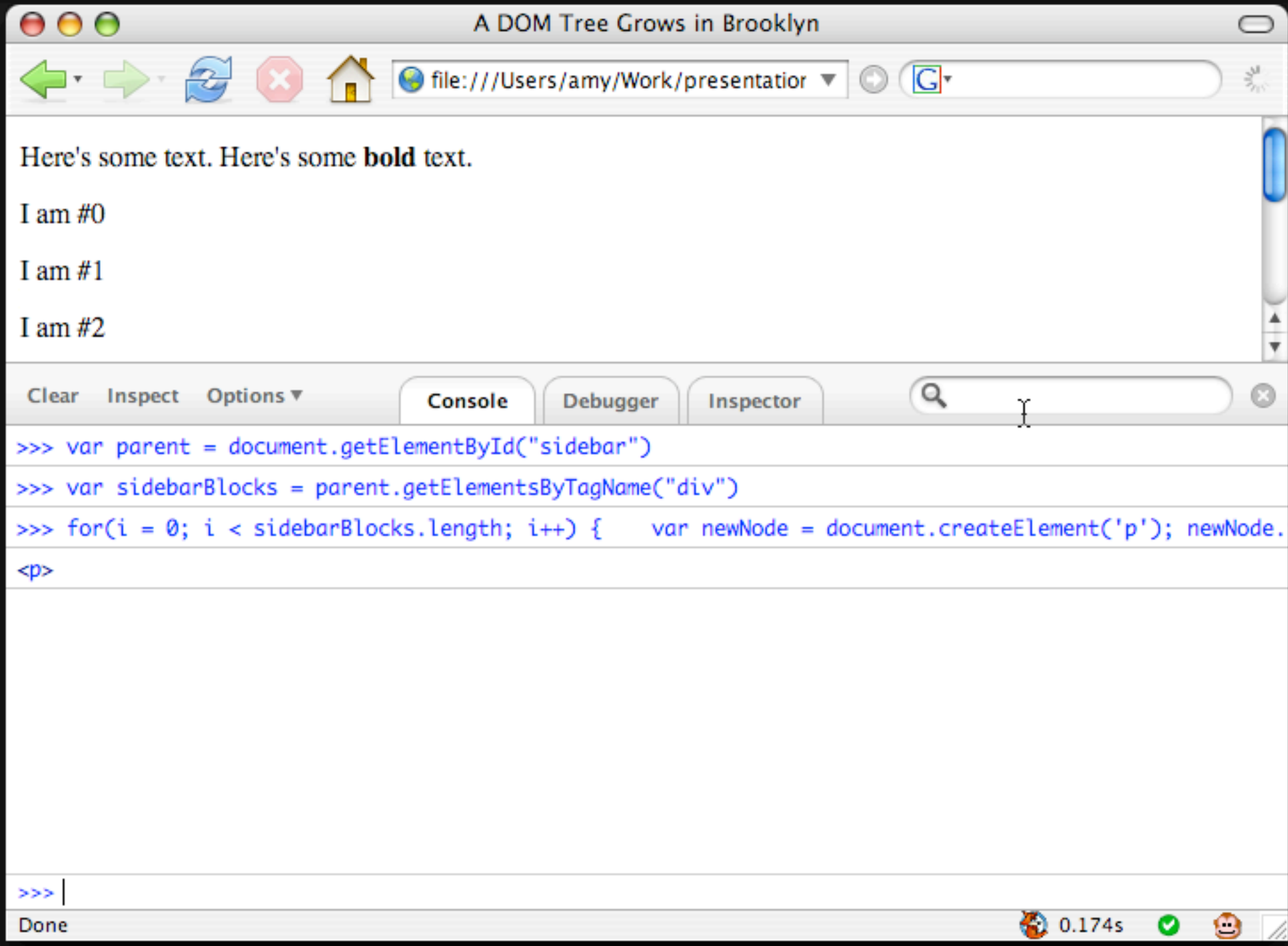

```
var node = document.getElementById("main")
// --> [object HTMLDivElement]

node.id
// --> "id"

node.tagName
// --> "DIV"

node.className
// --> "box"
```

finding DOM elements by tagName


```
var parent = document.getElementById("sidebar")
// --> [object HTMLDivElement]

var sidebarBlocks = parent.getElementsByTagName("div")
// --> [ div, div, div ]
```

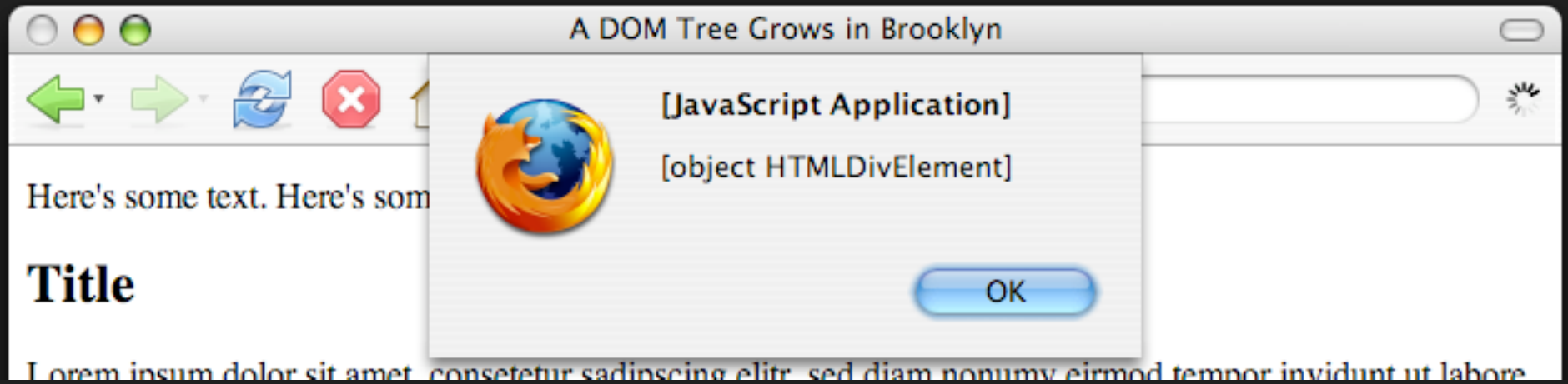
creating new DOM nodes


```
var parent = document.getElementById("sidebar")
// --> [object HTMLDivElement]

var sidebarBlocks = parent.getElementsByTagName("div")
// --> [ div, div, div ]

for(i = 0; i < sidebarBlocks.length; i++) {
    var newNode = document.createElement('p')
    newNode.innerHTML = "I am #"+i
    sidebarBlocks[i].appendChild(newNode)
}
```


A DOM Tree Grows in Brooklyn
file:///Users/amy/Work/presentatior
Here's some text. Here's some bold text.
I am #0
I am #1
I am #2
Clear
Inspect
Options
Console
Debugger
Inspector
>>> var parent = document.getElementById("sidebar")
>>> var sidebarBlocks = parent.getElementsByTagName("div")
>>> for(i = 0; i < sidebarBlocks.length; i++) { var newNode = document.createElement('p'); newNode.
<p>
>>>
Done
0.174s

embedding javascript

```
            <p>
                Duis autem vel eum iriure dolor in hendrerit in vulputate
                velit esse molestie consequat vel illum dolore eu feugiat.
            </p>
        </div>
    </div>
    <script>
        alert(document.getElementById("sidebar"))
    </script>
</body>
</html>
```



Prototype

```
Object.extend(RealClass, AbstractClass)
Object.extend(RealClass, {
  anotherMethod: function() {}
 }
)
```

```
// before:
document.getElementById('content')
// after:
$('content')

// <span class="title">title!</span>
// <span class="title">title 2!</span>
var titles = $$('span.title')
```

```
"<b>I'm a tag!</b>".stripTags()
// I'm a tag!

"<b>I'm a tag!</b>".escapeHTML()
// &lt;b&gt;I'm a tag!&lt;/b&gt;

[5,6,7,'peach',9].without('peach',7)
// [5,6,9]
```

```
var n = 10;
n.times(function(index){
    alert(index);
});
```

```
['one','two','three'].each (
    function(num,index) {
        msg = num+" in position "+index
    }
)
```

```
Event.observe('dropbox','mouseover',
              function(){}, true);

document.getElementsByClassName(
    'draggables')
```

```
new Ajax.Updater('ajax_result',
                 '/posts/1/comment');
```

```
new Ajax.Updater('ajax_result',
        '/posts/1/comment',
        onLoading: function() {
            Element.show('spinner')
        },
        onComplete: function() {
            Element.hide('spinner')
        }
 );
```

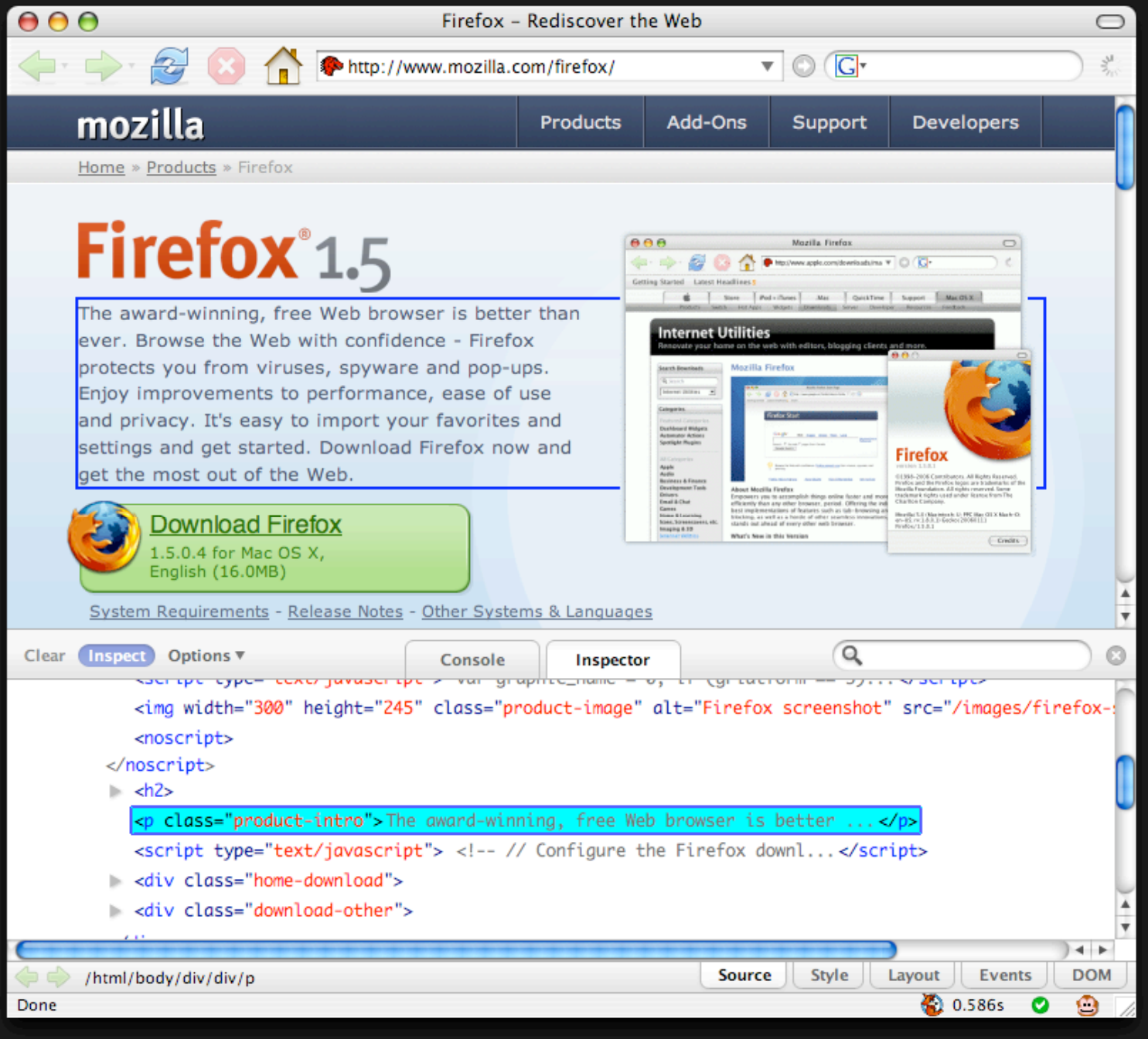
Development & Debugging

FireFox

your long-term best bud

FireBug

your new soulmate

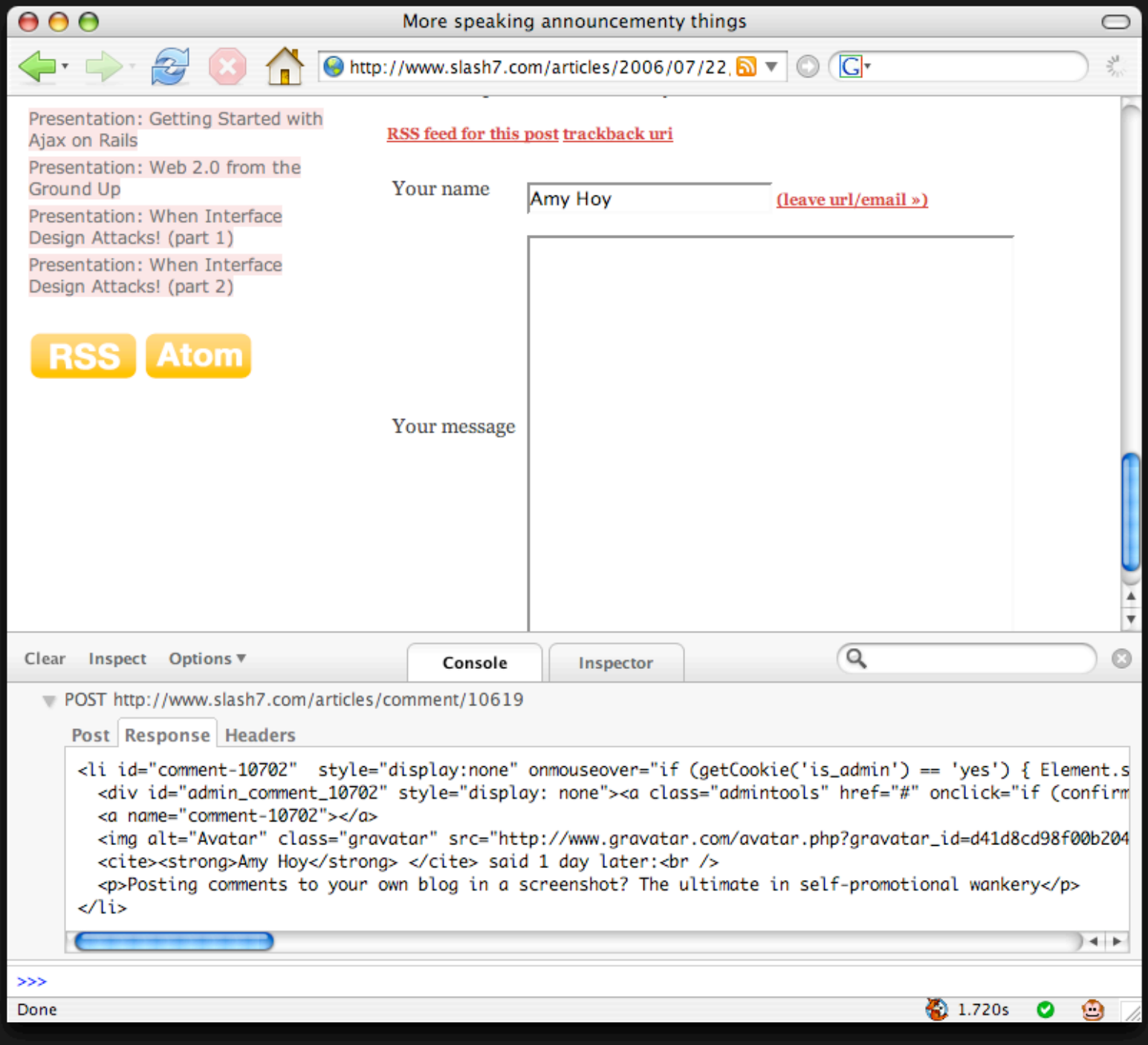


inspecting DOM elements: source

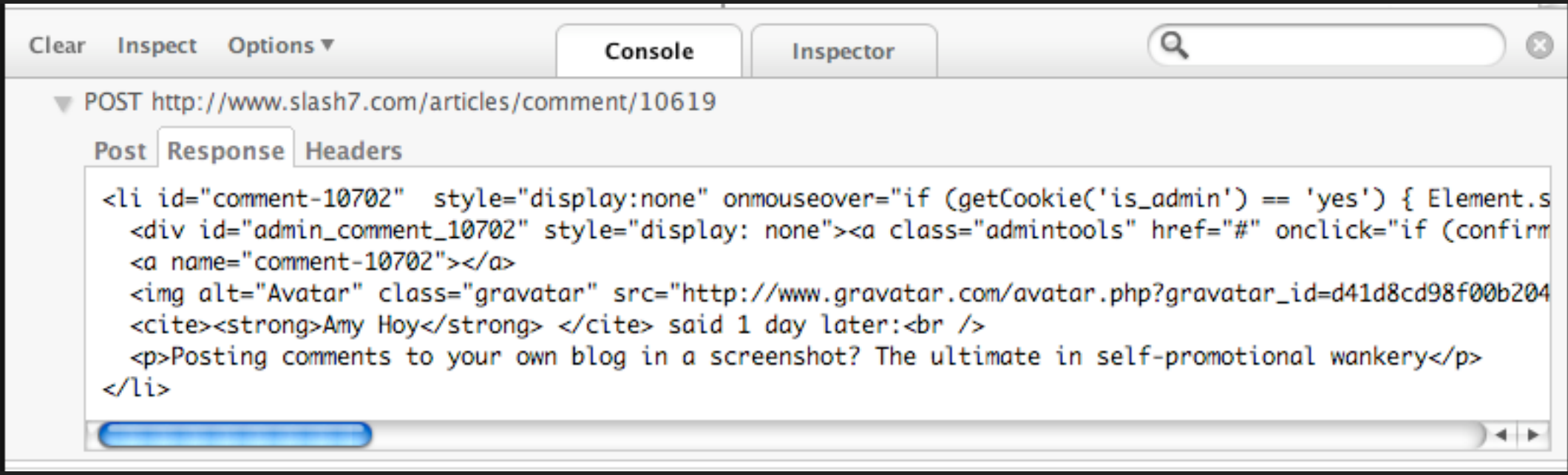
inspecting DOM elements: style

inspecting DOM elements: live events

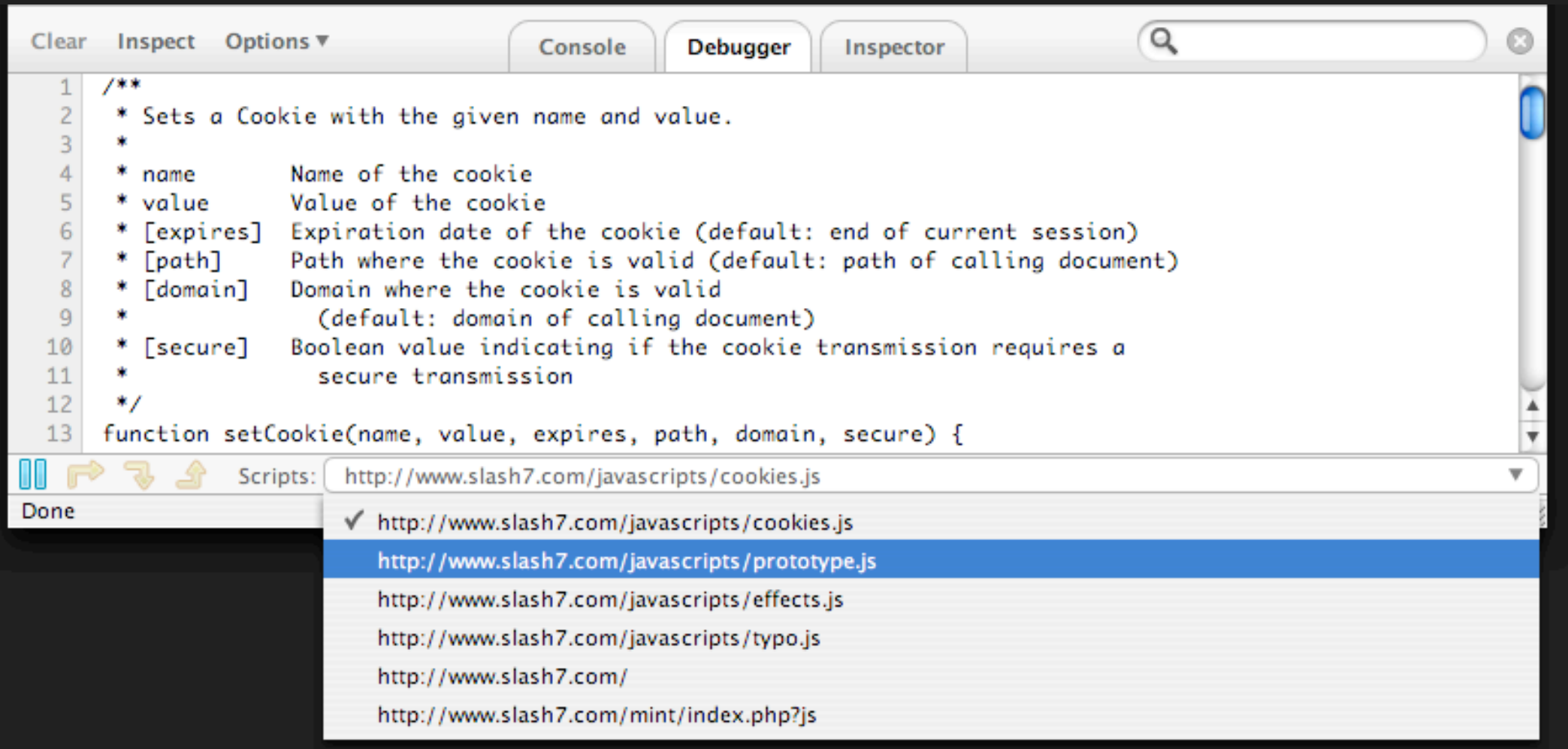
console: errors and filtering things to show



console: logging / inspecting AJAX requests



console: logging / inspecting AJAX requests



the step-thru debugger: new in FireBug .4

breakpoints

step-through debugging

flexible console logging

built-in profiling & testing functionality

live DOM inspector / editor

~~the ability to send email~~

Javascript Shell

no snazzy logo

www.squarefree.com/shell/shell.html

Other Tools & Libraries

JSUnit

MochiKit

jQuery

moo.ajax

TrimPath

moo.fx

Behavior & event:Selectors

Scriptaculous (fx)

moo.fx

Links for You

http://www.slash7.com/

look for the list of presentations under the "Goodies" sidebar