

# How DRY impacts JavaScript performance //

## Faster JavaScript execution for the lazy developer

---

Mathias Bynens – Velocity Europe, November 2011



**@mathias**

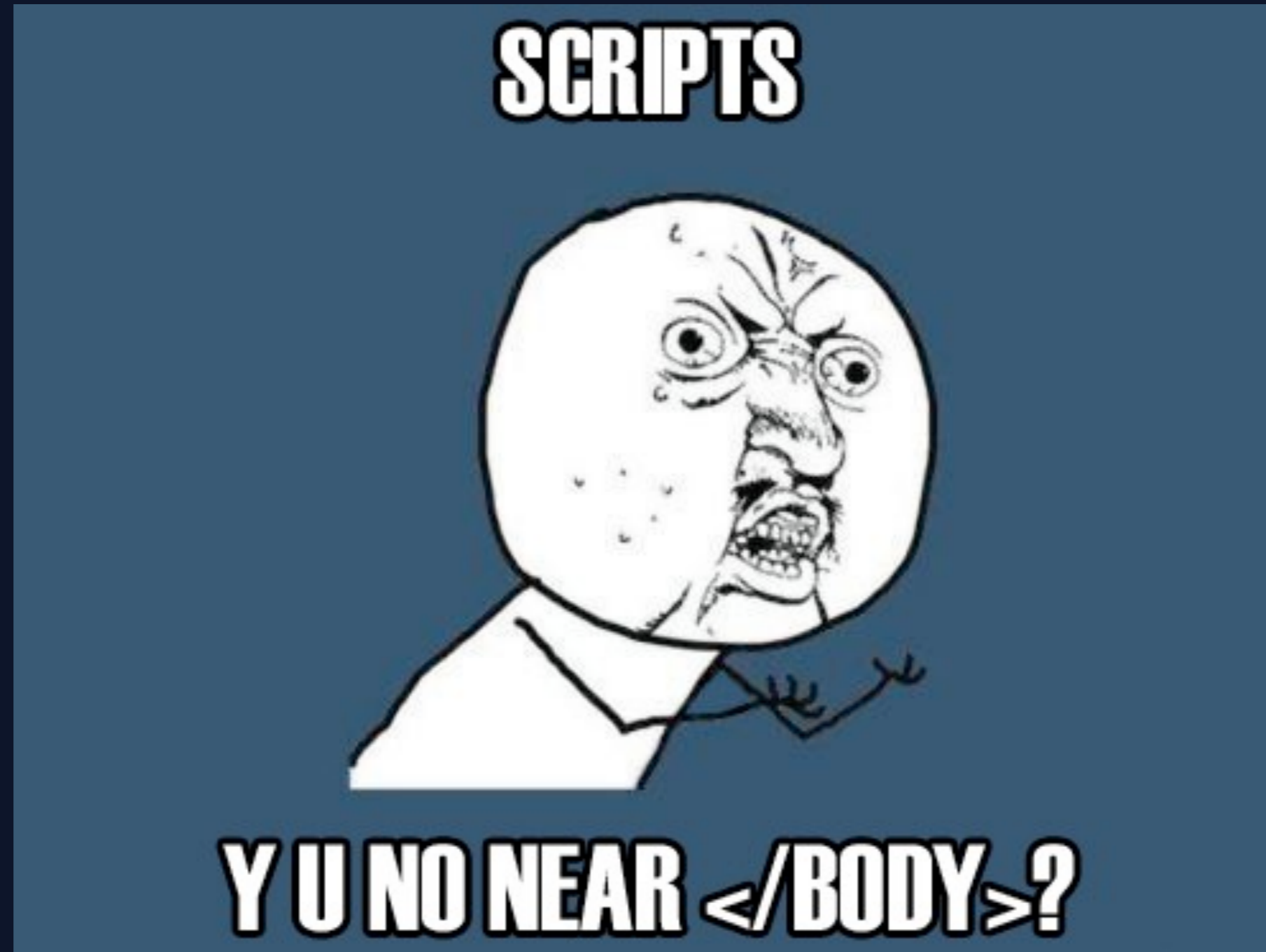
# JavaScript & performance

---

Rule #1: nothing to do with JS

# JavaScript & performance

---



# JavaScript & performance

---

What about  
the actual **run-time performance**  
on the **client side**?



**DRY**



**WET**

**“DRY leads to readable,  
maintainable code”**

# DRY JavaScript improves performance

...if you do it right

**So, where to avoid  
repetition?**

# What's slow in JavaScript?

---

# What's slow in JavaScript?

---

## 1. The DOM

# What's slow in JavaScript?

---

1. The DOM

2. Function calls

# What's slow in JavaScript?

---

1. The DOM

2. Function calls

3. Lookups

# DOM manipulation

---

```
// Create the element in memory
```

```
var e1 = document.createElement('p');
```

```
// Insert the element into the DOM
```

```
document.body.appendChild(e1);
```

# DOM manipulation

---

```
<body>
```

```
...
```

```
<div>
```

```
  <p></p>
```

```
</div>
```

```
</body>
```

# DOM manipulation

---

```
var div = document.createElement('div'),  
    p = document.createElement('p');
```

```
// Bad
```

```
document.body.appendChild(div);
```

```
div.appendChild(p);
```

# DOM manipulation

---

```
var div = document.createElement('div'),  
    p = document.createElement('p');
```

```
// Better
```

```
div.appendChild(p);
```

```
document.body.appendChild(div);
```

# DOM manipulation

---

<body>

...

<p></p>

<p></p>

<p></p>

<p></p>

</body>

# DOM manipulation

---

```
var p = document.createElement('p'),  
    i = 4;  
  
while (i--) { // Add four <p> elements  
    document.body.appendChild(p.cloneNode(false));  
}
```

# DOM manipulation

---

```
var frag = document.createDocumentFragment(),  
    p = document.createElement('p'),  
    i = 4;
```

```
while (i--) { // Add four <p> elements  
    frag.appendChild(p.cloneNode(false));  
}
```

```
document.body.appendChild(frag);
```

# Function calls

---

```
// Function declaration
```

```
function foo(bar) {  
    return bar;  
}
```

```
// Function call
```

```
foo('something');
```

# Function calls

---

```
alert('foo');
```

```
document.getElementById('foo');
```

```
$('#foo');
```

# Function calls

---

```
$('.foo').show();
```

```
// other stuff...
```

```
$('.foo').hide();
```

# Function calls

---

```
var $foo = $(' .foo');
```

```
$foo.show();
```

```
// other stuff...
```

```
$foo.hide();
```

# Function calls

---

```
var $foo = $(' .foo' ).show();  
// other stuff..  
$foo.hide();
```

# Property lookups

---

```
var obj = {  
  'x': 42,  
  'y': {  
    'foo': 'bar'  
  }  
};
```

```
obj.x; // 42
```

```
obj.y.foo; // 'bar'
```

# Property lookups

---

`document.title`

`dojo.query(...)`

`YAHOO.util.Dom.get(...)`

# Property lookups

---

```
var foo = YAHOO.util.Dom.get('foo'),  
    bar = YAHOO.util.Dom.get('bar'),  
    baz = YAHOO.util.Dom.get('baz'),  
    qux = YAHOO.util.Dom.get('qux');
```

# Property lookups

---

```
var get = YAHOO.util.Dom.get,  
    foo = get('foo'),  
    bar = get('bar'),  
    baz = get('baz'),  
    qux = get('qux');
```

# Array item lookups

---

```
var elems = document.getElementsByTagName('p'),  
    length = elems.length;
```

```
while (length--) {  
    if (elems[length].className == 'foo') {  
        // do something with elems[length]  
        elems[length].innerHTML = 'LOLWAT';  
    }  
}
```

# Array item lookups

---

```
var elems = document.getElementsByTagName('p'),  
    length = elems.length,  
    elem;
```

```
while (length--) {  
    elem = elems[length];  
    if (elem.className == 'foo') {  
        // do something with elem  
        elem.innerHTML = 'LOLWAT';  
    }  
}
```

# Scope lookups

---

```
var foo = 42;
```

```
foo; // no scope lookup
```

# Scope lookups

---

```
var foo = 42;  
(function() {  
    foo; // one scope lookup  
})();  
// IIFE – see http://mths.be/iife
```

# Scope lookups

---

```
var foo = 42;  
(function() {  
  (function() {  
    foo; // two scope lookups  
  }());  
}());
```

# Scope lookups

---



# Scope lookups

---

```
var foo = 42;  
(function(foo) {  
  (function(foo) {  
    foo; // ZOMG, no scope lookups!!1  
  })(foo);  
})(foo);
```

# Scope lookups

---



# Scope lookups

---

```
(function() {  
  // every time you use `window`  
  // or `document` here  
  // that's a scope lookup  
})();
```

# Scope lookups

---

```
(function() {  
    var doc = document,  
        win = window;  
    // lookup once, then cache  
})();
```

# Scope lookups

---

```
(function(win, doc) {  
  // use `win` and `doc` here  
  // no scope lookups  
  // no performance penalty!  
})(this, document);
```

# Recap: what's slow in JavaScript?

---

# Recap: what's slow in JavaScript?

---

## 1. The DOM

# Recap: what's slow in JavaScript?

---

1. The DOM

2. Function calls

# Recap: what's slow in JavaScript?

---

1. The DOM

2. Function calls

3. Lookups

Especially when used inside...

---

# Especially when used inside...

---

- Loops

# Especially when used inside...

---

- Loops
- Intervals

# Especially when used inside...

---

- Loops
- Intervals
- Handlers for events that fire frequently

# It happens to the best!

---

// Don't do this:

```
$(window).scroll(function() {  
    $('.foo').something();  
});
```

# It happens to the best!

---

// Don't do this:

```
$(window).scroll(function() {  
    $('.foo').something();  
});
```



# It happens to the best!

---

// Don't do this:

```
$(window).scroll(function() {  
    $('.foo').something();  
});
```



// See <http://mths.be/azs>

`typeof performance !== 'the whole story'`

tips & tricks  
*(not really)*

# New objects

---

```
var obj = new Object();
```

```
obj.x = 42;
```

```
obj.y = 'foo';
```

```
obj.z = false;
```

# New objects

---

```
var obj = {  
  'x': 42,  
  'y': 'foo',  
  'z': false  
};
```

# New arrays

---

```
var arr = new Array();
```

```
arr.push(42);
```

```
arr.push('foo');
```

```
arr.push(false);
```

# New arrays

---

```
var arr = [  
  42,  
  'foo',  
  false  
];
```

# Avoid switch

---

```
switch(foo) {  
  case 'alpha':  
    // do X  
    break;  
  case 'beta':  
    // do Y  
    break;  
  default:  
    // do Z  
    break;  
}
```

# Avoid switch

---

```
var switchObj = {
  'alpha': function() {
    // do X
  },
  'beta': function() {
    // do Y
  },
  '_default': function() {
    // do Z
  }
};
(switchObj.hasOwnProperty(foo) && switchObj[foo] || switchObj._default)(args);
```

# Don't use jQuery for everything

---

```
$('.foo').click(function() {  
    $(this).prop('id');  
    // same as this, before jQuery 1.6:  
    // $(this).attr('id');  
  
    // also `href`, `checked`, `value`...  
});
```

# Don't use jQuery for everything

---

```
$('.foo').click(function() {  
    this.id;  
    this.href;  
    this.checked;  
    this.value;  
    // etc.  
});
```

# jQuery document ready

---

```
$(document).ready(function() {  
    // teh coads  
});
```

# jQuery document ready

---

```
$(document).ready(function() {  
    // heh  
});
```

# jQuery document ready

---

```
$.fn.ready(function() {  
    // not pretty, but fastest solution  
});
```

# jQuery document ready

---

```
$(function() {  
    // moar sexy, but slower  
});
```

# jQuery document ready

---

```
(function() {  
    // move <script>s to the bottom  
    // and just use an IIFE*  
})();
```

// \* unless you use .appendChild() / .innerHTML on document.documentElement or document.body: <http://mths.be/ieoa>

# jQuery collection size

---

```
$('.foo').size(); // NO.
```

# jQuery collection size

---

```
// jQuery source:
```

```
$.fn.size = function() {  
    return this.length;  
};
```

```
// ...so, just use:
```

```
$('.foo').length;
```

# Use context

---

```
$('#foo .bar').addClass('baz');
```

```
$('#foo .qux').hide();
```

```
$('#foo input').removeClass('wut');
```

# Use context

---

```
var $foo = $('#foo');  
$('.bar', $foo).addClass('baz');  
$('.qux', $foo).hide();  
$('.input', $foo).removeClass('wut');
```

```
this.location = 'http://jsperf.com/'
```

<http://jsperf.com/browse/mathias-bynens>

# Questions?

@mathias