Unpacking the Black Box

Benchmarking JS Parsing and Execution on Mobile Devices

Daniel Espeset
@danielespeset
Resources for this talk

talks.desp.in/unpacking-the-black-box
Etsy

Frontend Infrastructure

Seth Walker  Daniel Espeset  Daniel Na  Takashi Mizohata
Etsy

Performance

Lara Hogan  Natalya Hoota  Jonathan Klein  Allison McKnight
>50% Mobile Traffic

10 Languages
Thinking about Mobile Performance
1. The Network
2. The Browser
degraded render performance
JavaScript initializing
jquery.js
parse → execute
How long does our JavaScript payload take to parse and execute, once it hits the browser?
To the lab!
Let's Add Some Timers!
Adding naive timers to our JS bundle

```javascript
1 var start = new Date().getTime();
- // ... bulk of your code
600 var duration = new Date().getTime() - start;
```
Results! **271ms** for our main shared JS bundle
This is awesome. Do this if you don't already.
A bunch of work happens before this runs.

```javascript
var start = new Date().getTime();
// ... bulk of your code
var duration = new Date().getTime() - start;
```
parse \rightarrow execute
Next questions

1. How long does it take to *parse*?
2. Can we measure this on every device in our lab?
What Are Our Options?
Remote Debugging
Details: Evaluate Script

Self Time: 223.591 ms
Start Time: 669 ms
Script: *jquery.js:1*

229.361 ms

223.591 ms Scripting (Self)
5.770 ms Loading
Pure JS Solutions
We could wrap timers around a `<script>` tag

```javascript
20 <script>var beforeJquery = new Date().getTime();</script>
21 <script src="jquery.js"></script>
22 <script>var took = new Date().getTime() - beforeJquery;</script>
```
This doesn’t work, it may include the network or exclude parse time.
Same idea, but in a loop

```javascript
var runs = 1000;
results = [];
while (runs--) {
    d.write(\`\`var start = Date.now();\`\`);
    d.write(\`\`<script src='jQuery.js'></script>\`\`);
    d.write(\`\`<script>\`\`);
    d.write(\`\`results.push(Date.now() - start);\`\`);
    d.write(\`\`var e = d.getElementById('test');\`\`);
    d.write(\`\`e.parentNode.removeChild(e);\`\`);
    d.write(\`\`</script>\`\`);
}
```
Carlos Bueno did this in 2010

“Measuring Javascript Parse and Load”
bit.ly/js-parse-and-load
Runtime is **complicated**

JIT Optimizations

Garbage Collection

bit.ly/js-parse-and-load
Our approach
Step 1. “window scoper” transforms implicit globals to explicit ones.

```javascript
1 var foo = {};  
2 (function() { 
    // ...
99 });

1 window.foo = {};  
2 (function() { 
    // ...
99 });
```
Step 2. wrap the contents in a named function

```
window.foo = {}; // ...
```

```
function sourceCode() {
    window.foo = {}; // ...
    ...
}
```
Step 3. turn the function into a string, eval and execute it

```javascript
var start = new Date().getTime();
eval("function sourceCode() { window.foo = {}; ... }");
var parse = new Date().getTime();
var parseTook = start - parse;
sourceCode();
var execTook = new Date().getTime() - parse;
```
Step 3. turn the function into a string, eval and execute it

```javascript
var start = new Date().getTime();
eval("function sourceCode() { window.foo = {}; ... }");
var parse = new Date().getTime();
var parseTook = start - parse;
sourceCode();
var execTook = new Date().getTime() - parse;
```
Step 3. turn the function into a string, eval and execute it

```javascript
1 var start = new Date().getTime();
2 eval("function sourceCode() { window.foo = {}; ... "});
3 var parse = new Date().getTime();
4 var parseTook = start - parse;
5 sourceCode();
6 var execTook = new Date().getTime() - parse;
```
results for jquery 1.8.2 + almond

Apple (Macbook Pro)

Chrome 31
RAM: 16GB CPU: 2.4GHz GPU: 1058MHz

36ms total
results for jQuery 1.8.2 + almond

Samsung Galaxy S3
Mobile Chrome 31
RAM: 1GB CPU: 1.4GHz GPU: 400MHz

319ms total
results for jquery 1.8.2 + almond

LG Optimus V (VM670)
Android Browser 4.0
RAM: 512MB  CPU: 600MHz  GPU: 128MHz

1389ms total
What does this measure?

1. How long to eval a big function with our code
2. How long to execute it
Assumptions

1. Eval is a reasonable proxy for `<script>`
2. This effectively separates parse and exec
Add timers to the top and bottom of the big wrapper function

```javascript
var start = new Date().getTime(), innerStart, innerEnd;
eval("function sourceCode() {
    innerStart = new Date().getTime();
    window.foo = {}; ...
}");
var execStart = new Date().getTime();
sourceCode();
var execEnd = new Date().getTime() - parse;
```
Add a timer to the top of the big wrapper function

```javascript
var start = new Date().getTime(), innerStart, innerEnd;
eval("function sourceCode() {
    innerStart = new Date().getTime();
    window.foo = {}; ...
}");
var execStart = new Date().getTime();
sourceCode();
var execEnd = new Date().getTime() - parse;
```
(what this looks like at runtime)

```javascript
var execStart = new Date().getTime();
(function sourceCode() {
    innerStart = new Date().getTime();
    window.foo = {};
    // ...
})();
var execEnd = new Date().getTime() - parse;
```

The time between these should be \(\approx 0\) ms
(what this looks like at runtime)

```javascript
var execStart = new Date().getTime();
(function sourceCode() {
    innerStart = new Date().getTime();
    window.foo = {};
    // ...
})();
var execEnd = new Date().getTime() - parse;
```

Chrome on Samsung Galaxy S3

50–100ms
Runtime is **complicated**

- JIT Optimizations
- Garbage Collection

Chrome on Samsung Galaxy S3

50–100ms
parse → execute
Step 3 REDUX: turn the function into a string, eval it

1. var start = new Date().getTime();
2. eval("var parse = new Date().getTime(); !function(){ ... }()" знаков);
3. var execTook = new Date().getTime() - parse;
4. var parseTook = start - parse;
Parse: the time between start, and the first line of the eval running.

1. `var start = new Date().getTime();`
2. `eval("var parse = new Date().getTime(); !function(){ ... }()"');`
3. `var execTook = new Date().getTime() - parse;`
4. `var parseTook = start - parse;`
Exec: the time between the first line of the eval, and the line after

```javascript
var start = new Date().getTime();
eval("var parse = new Date().getTime(); !function(){ ... }()"可谓);
var execTook = new Date().getTime() - parse;
var parseTook = start - parse;
```
DeviceTiming
DeviceTiming
Server
DEMO TIME!
demo.desp.in

The backbone example from todomvc.com
Optimizations

Cutting The Mustard

Going Vanilla

You could be blocked on CSS
Etsy

codeascraft.com

etsy.com/careers