Real World JS Code

From Alexa Top 10k
About the Dataset

• Alexa Top 10,000 sites
• Hit URL, downloaded all linked and inline scripts
• Adult websites not included (blocked by proxy, can workaround for future analysis if required)
• 122,491 .js files represented (1.14GB)
• Sent non-IE UA string to avoid IE-specific code.
Function in Block (F-I-B)

- 185/10,000 sites (1.85%) are potentially broken by proposed block-scoped decls semantics
  - See here for some preliminary results with detailed code samples: https://mail.mozilla.org/pipermail/es-discuss/2012-December/027419.html

- For the most part, code depends on behavior that is currently interoperable across major browsers.

- Some libraries have issues:
  - 42 breaks are due to Thickbox 3.0 library (prev/next with keyboard are broken)
  - 13 breaks are due to Qtip 1.0 library (tooltips will not disappear after being shown)
  - 8 due to OpinionLab library

Function in Block (F-I-B), Continued

- Functions declared in blocks are extremely common: 1,674/10,000 (16.75%) of sites contain declarations in block scope.
  - 185 (11%) of these sites would be broken by proposed block-scoped decls semantics.

- Immediate parent block
  - Vast majority of declarations are immediately inside if statements, followed by try statements.
  - Examined a random sample of 10 scripts with decls in for, for-in, and switch. 7 decls were simple helper methods. 2 were event callbacks. 1 was an array sort callback. In all cases care was clearly taken to avoid attempting to closure capture loop var.

<table>
<thead>
<tr>
<th>Node Type</th>
<th>Number of Decls</th>
</tr>
</thead>
<tbody>
<tr>
<td>IfStatement</td>
<td>6135</td>
</tr>
<tr>
<td>TryStatement</td>
<td>2042</td>
</tr>
<tr>
<td>ForStatement</td>
<td>19</td>
</tr>
<tr>
<td>ForInStatement</td>
<td>17</td>
</tr>
<tr>
<td>SwitchCase</td>
<td>15</td>
</tr>
<tr>
<td>BlockStatement</td>
<td>6</td>
</tr>
<tr>
<td>CatchClause</td>
<td>1</td>
</tr>
<tr>
<td>LabeledStatement</td>
<td>1</td>
</tr>
</tbody>
</table>
Function in Block (F-I-B), Continued

• Same breakdown, considering every parent up until a Function or Program body.

<table>
<thead>
<tr>
<th>Node Type</th>
<th>Number of Decls</th>
</tr>
</thead>
<tbody>
<tr>
<td>IfStatement</td>
<td>8126</td>
</tr>
<tr>
<td>TryStatement</td>
<td>2055</td>
</tr>
<tr>
<td>ForStatement</td>
<td>40</td>
</tr>
<tr>
<td>SwitchStatement</td>
<td>37</td>
</tr>
<tr>
<td>SwitchCase</td>
<td>37</td>
</tr>
<tr>
<td>ForInStatement</td>
<td>22</td>
</tr>
<tr>
<td>BlockStatement</td>
<td>10</td>
</tr>
<tr>
<td>CatchClause</td>
<td>1</td>
</tr>
<tr>
<td>LabeledStatement</td>
<td>1</td>
</tr>
</tbody>
</table>
Duplicate Parameter Names

• 7 out of 10,000 (0.07%)
• 3 / 7 the duplicated parameter is unused
• None of the identifiers are _.
• No libraries affected
• Convention of using _ to ignore unused parameters appears on 178 scripts (118 unique) across 163 sites
  • Most of these results are from Jquery
  • Presumably there is a desire to use this pattern
• Minification has a major impact on this data ( _ identifiers get “minified” to other identifiers)
Strict Mode & Other Prologs

• 404/10,000 (4.04%) of sites contain some code in strict mode
• 75/10,000 (0.75%) of sites contain “use strict” outside of the directive prolog. Sample of 10 scripts shows this is universally due to concatenation issues.
• Of 1301 strict mode directives, 96% are in function expressions, 3% in programs, and 1% in function declarations.
• 369 sites use "use strict", vs. 105 use 'use strict'
• Only other valid directive usage is “$:nomunge” on 31 sites (0.31%)
• 36/10,000 (0.36%) use invalid directives. These including code comments, html snippets, and other misc. strings.
Const Usage

• 1/10,000 (0.01%) of sites use const (instaforex.com, rank 2047).
• Code is, effectively:

```javascript
var define = (function() {
    try {
        eval('const x = 1');
        return function(name, value) {
            // create script body containing "const " + name + "=" + value
        }
    } catch(e) {
        if(this.execScript) {
            return function(name, value) {
                execScript("const " + name + "=" + value, 'vbscript');
            }
        }
    }
})();
```
let[x] = ? (destructuring breaking change)

- Not found at all in this dataset.
Var decl in catch clause with same name

• 1/10,000 (0.01%) of sites have a var decl of the same name as the catch clause parameter (marca.com, rank 351).
• Code is try { /* lots of code */ } catch(err) { var err; }.