Proposal Update: Function Implementation Hiding

Stage 2 • Michael Ficarra • 73rd Meeting of TC39

Goal

For cases where functions fail to act as an encapsulation boundary, provide a straightforward way for JavaScript authors to rectify that.

Considered Use Cases: Polyfills

A high-fidelity polyfill that is indistinguishable from a built-in.

- present in stack traces
  - with name
  - without file name / URL
  - without line / column number
- source text is unavailable
- name / length properties auto-populated but can be overwritten
- uses `"hide source"` directive

Considered Use Cases: Libraries (1)

Libraries that do not want consumers of exposed functions/methods to rely on what many consider to be implementation details and not an intentional part of their API.

- present in stack traces
  - with name
  - without file name / URL
  - without line / column number
- source text is unavailable
- name / length properties auto-populated but can be overwritten
- uses `"hide source"` directive

Considered Use Cases: Libraries (2)

Libraries that would like to factor out helper functions without changing observable aspects of their API.

- not present in stack traces
- source text availability doesn't matter, since it is unreachable from the outside
- name / length don't matter, since it is unreachable from the outside
- uses `"sensitive"` directive

Considered Use Cases: Security

Any security-sensitive use cases that rely on the confidentiality of the source text, the local bindings, or the calling behaviour of a function.

- not present in stack traces
- source text is unavailable
- name / length properties arguably don't affect source text confidentiality, though can be overwritten
- uses `"sensitive"` directive

Proposal Content

1. "hide source" directive
    - Function.prototype.toString result matches NativeFunction
        - accomplishes two goals: appears built-in and doesn't expose original source text
    - In stack traces, no accompanying attribution (file / URL) or position (line / column)
2. "sensitive" directive
    - Function.prototype.toString result matches NativeFunction
    - Never appears in stack traces

Changes Since Last Time

- changed "hide implementation" to "hide source"
- many README improvements
  - added examples of behaviour
  - clarified intended purpose of "sensitive" directive
  - clarified that no restrictions for privileged/external code
- opened PR to tc39/ecma262 with full spec text
  - approved by Mike Samuel and Jordan
  - not reviewed by Yehuda
  - requests for changes from Waldemar and Allen

Discussion Points

- name / length properties: don't auto-populate?
  - if not, what should the values be? empty string and zero? null? undefined? not present?
- Function.prototype.toString censorship not propagating upward
- runtime introspection vs security / encapsulation trade-off
- bikeshed directive names: "confidential" instead of "sensitive"?
- stage 3 blocker: Waldemar/Allen's review of spec text

Sample Code 1

```
function f(x) {
  return function g() {
    'hide source';
    return x;
  };
}

f.toString();
```

Sample Code 2

```
function f(x) {
  'hide source';
  return eval('(function g() {\
    return x;\
  })');
}

f().toString();
```