

Dynamic Code Analysis for JavaScript

@ariyahidayat

Feb 12, 2014

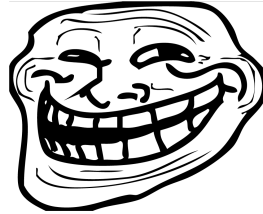


/usr/bin/whoami



SH-PE

shapesecurity.com



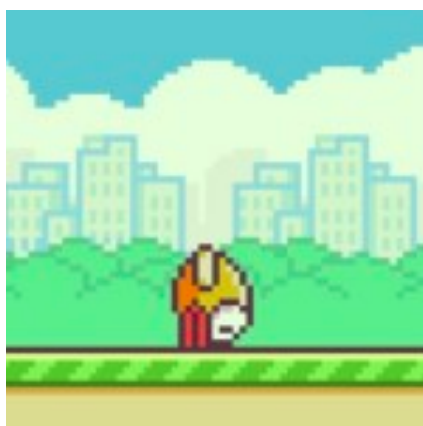
“Software Provocateur”



PhantomJS



Esprima

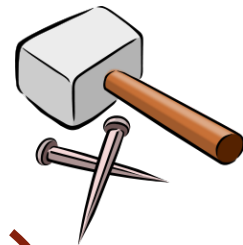




“YOU SHALL NOT PASS!”

— *Darth Vader*

Utility Belt



Parser

+

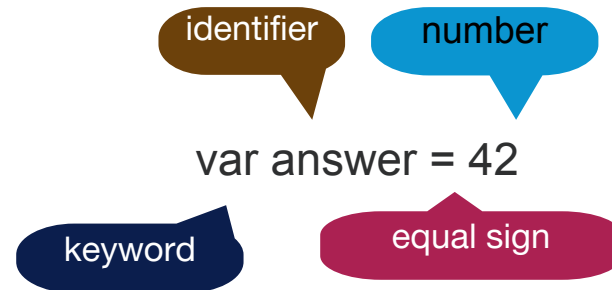
Code
Generator

Function
Instrumentation

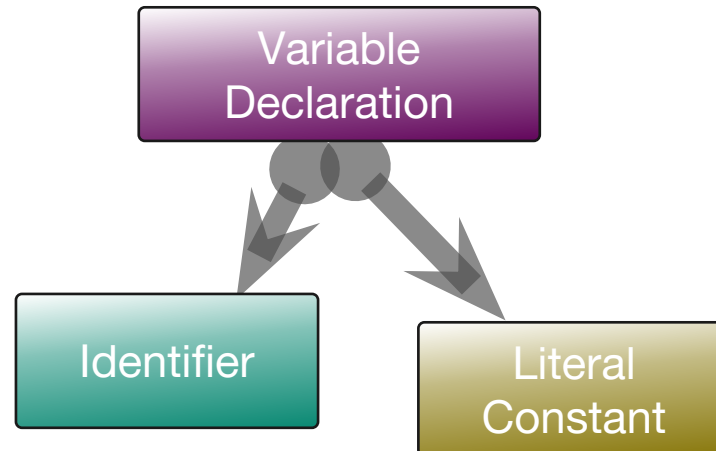
Code
Coverage

Syntax Parser

Tokenization → Tokens



Parsing → Syntax Tree



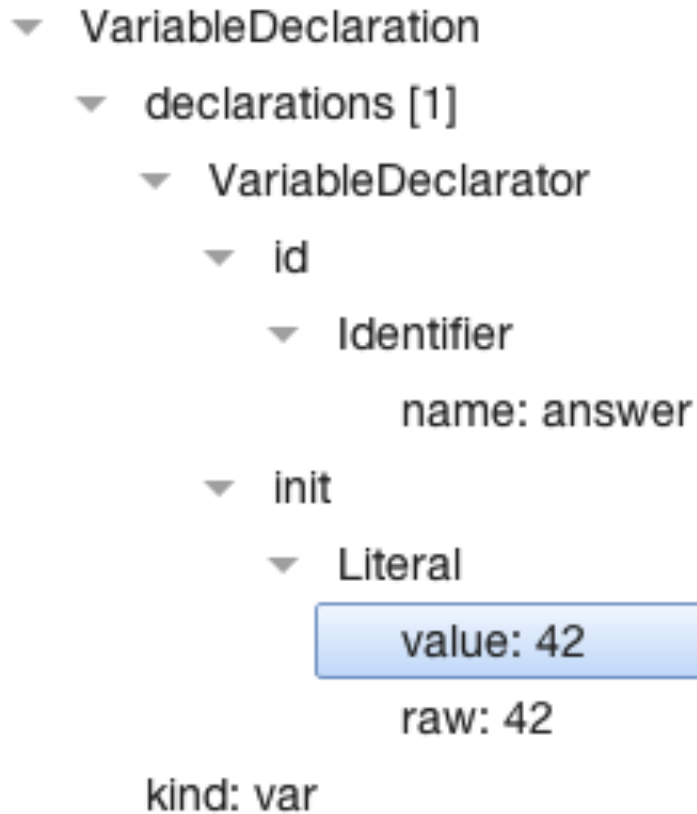
Syntax Tree

```
var answer = 42;
```

Terms → ECMAScript 5.1 Specification

```
{
  type: "Program",
  body: [
    {
      type: "VariableDeclaration",
      declarations: [
        {
          type: "VariableDeclarator",
          id: {
            type: "Identifier",
            name: "answer"
          },
          init: {
            type: "Literal",
            value: 42,
            raw: "42"
          }
        }
      ]
    }
  ],
  kind: "var"
}
```

Syntax Visualization



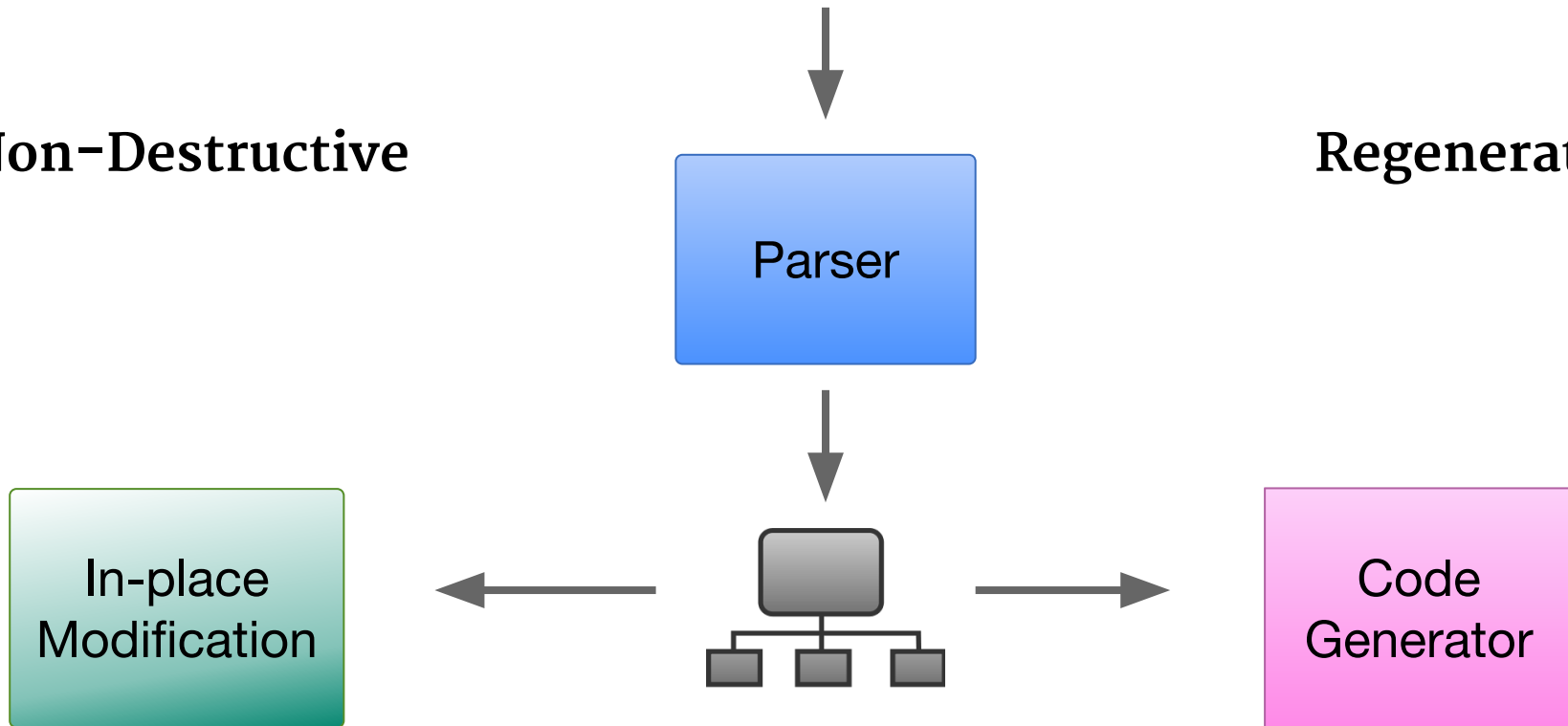




Source Transformation

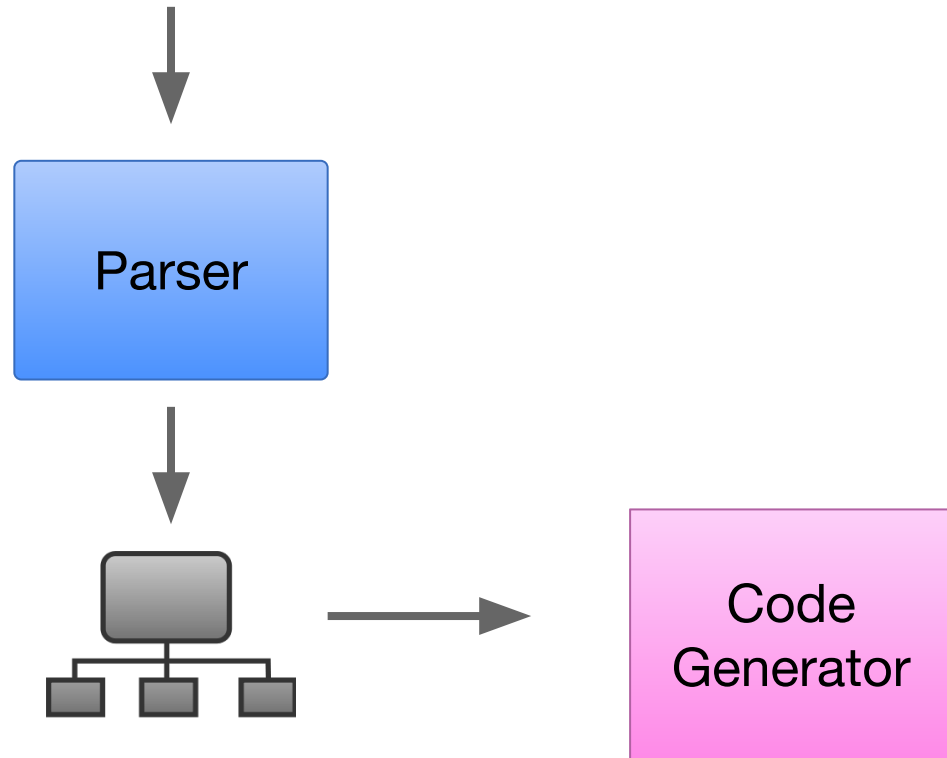
Non-Destructive

Regenerative



Code Coverage

Regenerative Transformation



Istanbul



Code Instrumentation

```
var answer = 42;  
alert(answer);
```



```
var __cov_1$4m7m$L464yvav5F$qhNA = __coverage__['hello.js'];  
__cov_1$4m7m$L464yvav5F$qhNA.s['1']++;  
var answer = 42;  
__cov_1$4m7m$L464yvav5F$qhNA.s['2']++;  
alert(answer);
```


Statement/Expression Coverage

```
function fibo(n) {  
  return (n < 2) ? n : fibo(n-1) + fibo(n-2);  
}
```

```
assert("fibo(1) must give 1", fibo(1) == 1);
```

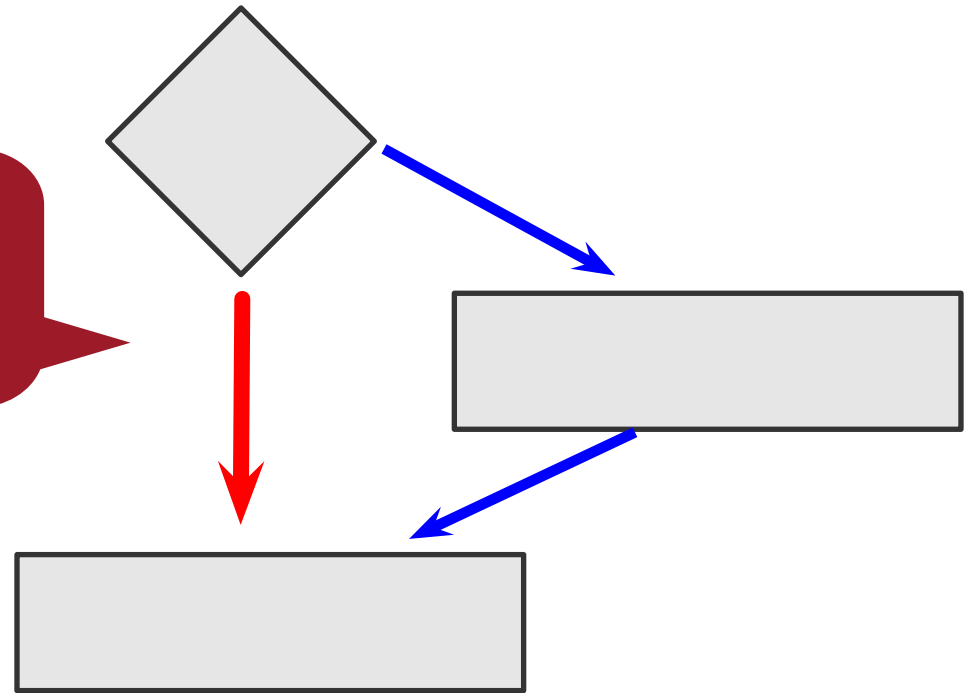
1	1	function fibo(n) {
2	1	return (n < 2) ? n : fibo(n-1) + fibo(n-2);
3		}
4		

Branch Coverage

```
function inc(p, q) {  
    if (q == undefined) q = 1;  
    return p + q/q;  
}  
assert("inc(4) must give 5", inc(4) == 5);
```

1	1	function inc(p, q) {
2	1	E if (q == undefined) q = 1;
3	1	return p + q/q;
4		}
5		

Does not catch the missing
code sequence



Input



Shortcut
possible?



Yes



No

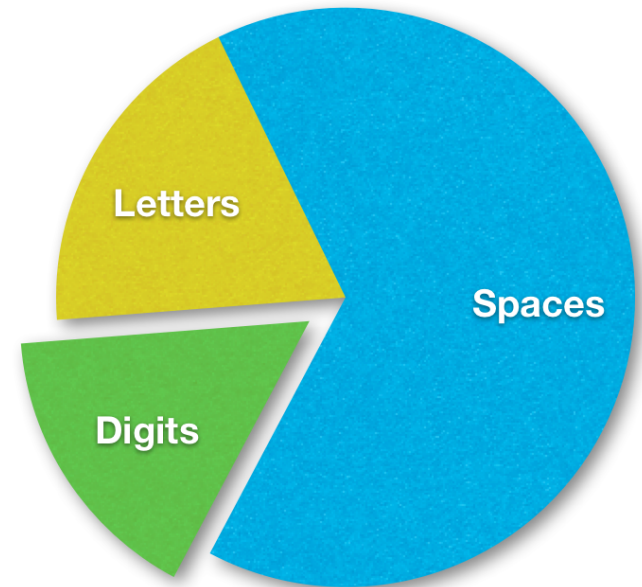


Profile-Guided Optimization

```
function isDigit(ch) {  
    return '0123456789'.indexOf(ch) >= 0;  
}
```



```
function isDigit(ch) {  
    return (ch !== ' ') && '0123456789'.indexOf(ch) >= 0;  
}
```



Istanbul, QUnit, Karma

```
test("sqrt", function() {
  deepEqual(My.sqrt(4), 2, "square root of 4 is 2");
});
```

1	1	var My = {
2		sqrt: function(x) {
3	1	if (x < 0) throw new Error("sqrt can't work on negative number");
4	1	return Math.exp(Math.log(x)/2);
5		}
6		};
7		

“If you think JSLint hurts
your feelings, wait until you
use Istanbul.”



@davglass

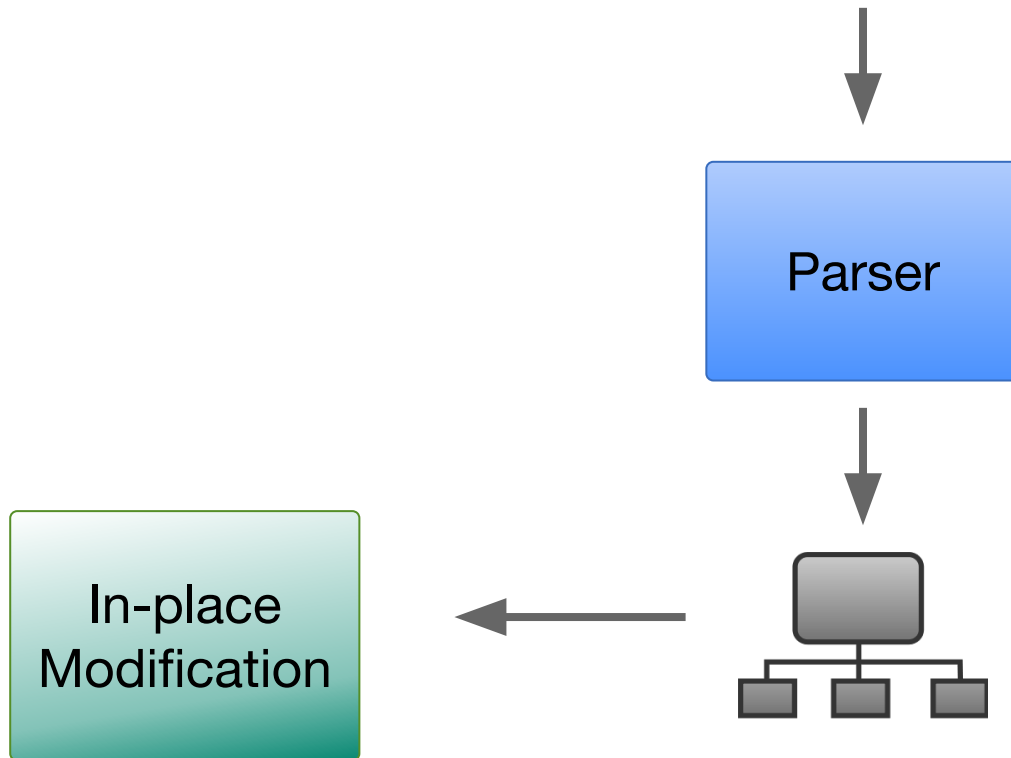


*...I gave you my heart
But the very next day you gave it away...*



Function Instrumentation

Non-Destructive Transformation



String Literal Quotes

```
console.log("Hello")
```



```
console.log('Hello')
```

Timing

Prepare the stopwatch
Mark the start and the end

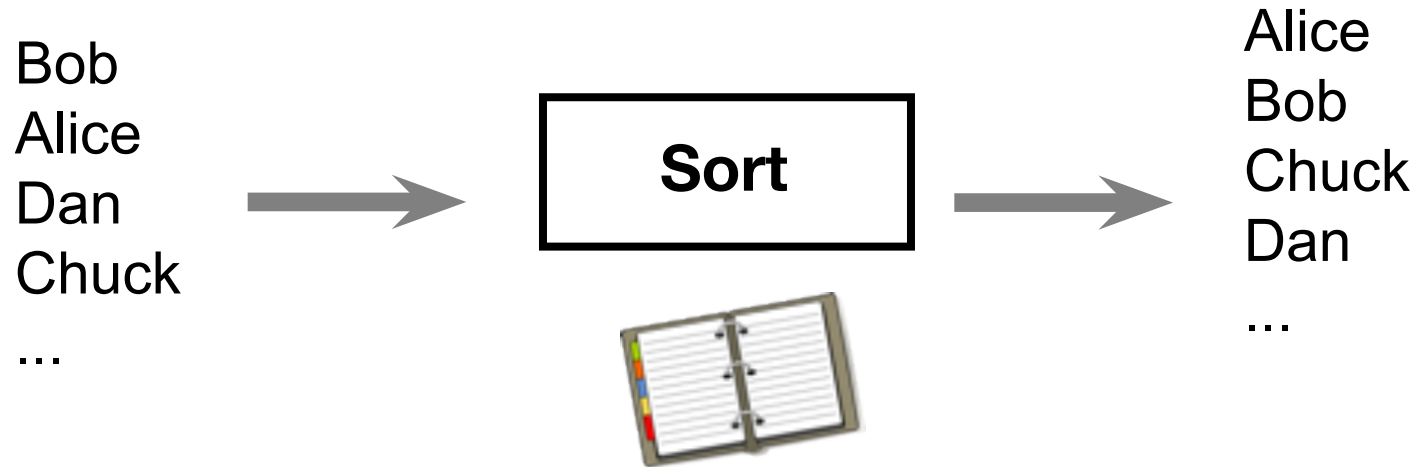
Sampling

Periodically check which function
is being executed

Tracing

Track all function calls and exits

Fast = Enough?



How's the speed?

2 ms to sort 10
contacts

Bubble Sort

???

```
Array.prototype.swap = function (i, j) {  
    var k = this[i]; this[i] = this[j]; this[j] = k;  
}  
  
function sort(list) {  
    var items = list.slice(0), swapped = false, p, q;  
    for (p = 1; p < items.length; ++p) {  
        for (q = 0; q < items.length - p; ++q) {  
            if (items[q + 1] < items[q]) {  
                items.swap(q, q + 1);  
                swapped = true;  
            }  
        }  
        if (!swapped) break;  
    }  
    return items;  
}
```

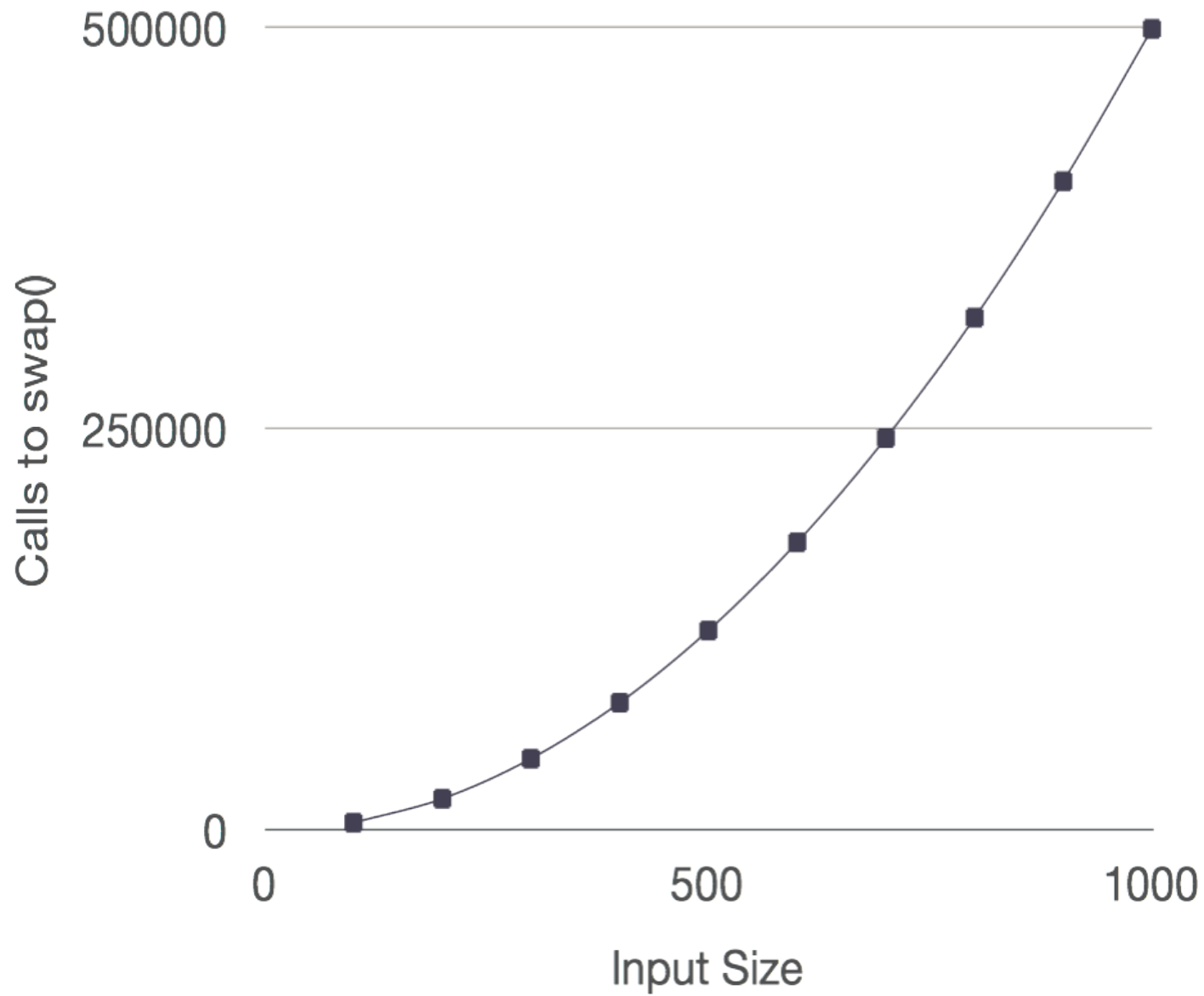


Run-time Complexity

```
Array.prototype.swap = function (i, j) {  
    var k = this[i]; this[i] = this[j]; this[j] = k;  
}
```

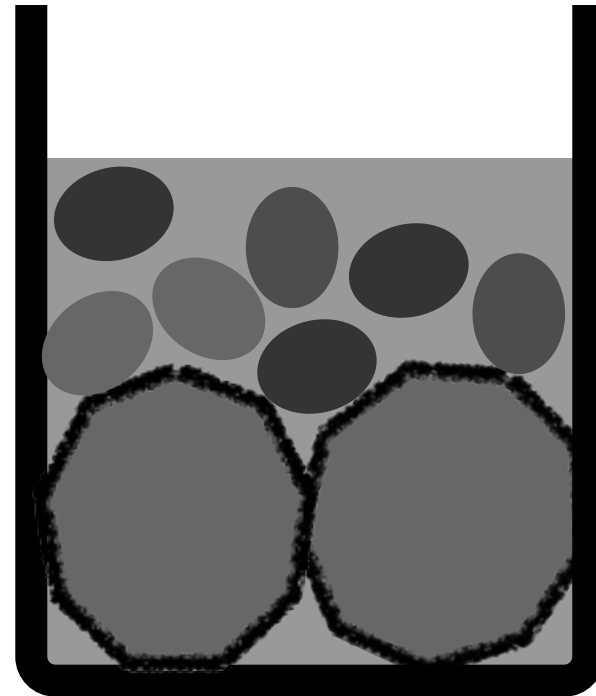
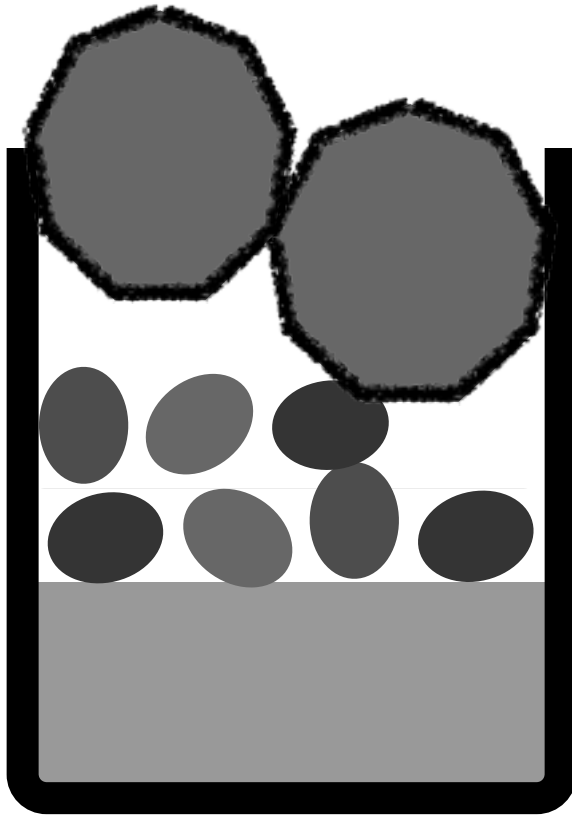


```
Array.prototype.swap = function (i, j) {  
Log({ name: 'Array.prototype.swap', lineNumber: 1, range: [23, 94] });  
    var k = this[i]; this[i] = this[j]; this[j] = k;  
}
```



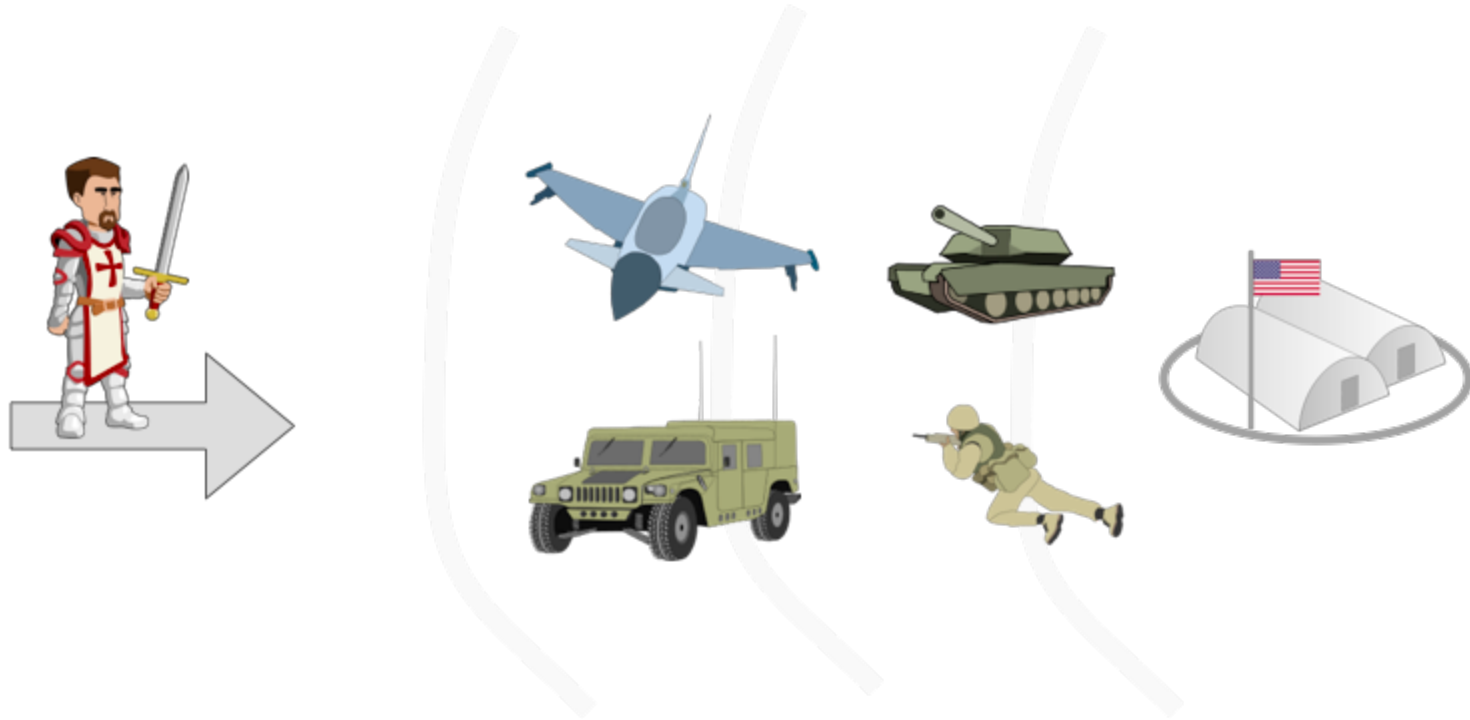
“The Big Rock”

“First Things First”, Steven Covey



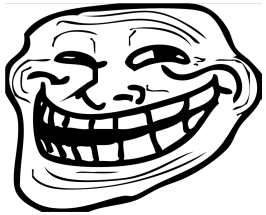
Layers of Defense

Being Defensive



Being Objective

Mr. Reviewer



Your code sucks!



Your code sucks!

Tests Madness

T D D

B D D



Coverage Thresholds

```
istanbul check-coverage --statement -5 --branch -3 --function 100
```

```
Tests: 19   Failures: 0
=====
Writing coverage object [/home/ariya/dev/esrefactor/coverage/coverage.json]
Writing coverage reports at [/home/ariya/dev/esrefactor/coverage]
=====

===== Coverage summary =====
Statements   : 98.13% ( 105/107 )
Branches     : 93.55% ( 58/62 )
Functions    : 100% ( 12/12 )
Lines        : 98.11% ( 104/106 )
=====

> istanbul check-coverage --branch -3

/usr/local/node/lib/node_modules/istanbul/lib/cli.js:31
    throw ex; // turn it into an uncaught exception
    ^
ERROR: Uncovered count for branches (4) exceeds threshold (3)
```



Tools separate us

Final Words



Dynamic code analysis for quality:

- Code coverage (statement, branch,...)
- Function instrumentation

Build, write, tweak code analysis tools

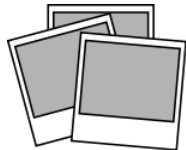
Thank You



@ariyahidayat



ariya.ofilabs.com



speakerdeck.com/ariya

