





# JSConf[2017].talk.by('Jan Krutisch').aka('@halfbyte')

```
eval(z='p=<"+"pre> /* *#####*/* */;for(y in n="zw24l6k\  
4e3t4jnt4qj24xh2 x/* #####*/* */42kty24wrt413n243n\  
*/*43iyb6k43pk7243nm*/*for(a in t=pars*#*/(e=x=r=[ ])for*#*/]>i;i+=.05)wi*#.05,0<cos(o=##*/-x/PI)&&(e[~*####*/sin(.5+y/7))####*/for(x=0;122>##*/[e[x++]+e[x++* */+(z=\\"+z.split*/join(B+B).split*/)+Q+"")//m1k")[x/2* */(/\\w/.test(S)&&"#03B");document.body.innerHTML=p+=B+"\\"+n"}setTimeout(z)' )//m1k\
```

```
eval(z='p=<"+"pre> /* *#####
4e3t4jnt4qj24xh2 x/* #####*#####
9h243pdxt41csb yz/*#####*#####
r24".split(4)){/* #####*#####
eInt(n[y],36)+/* #####*#####
(r=!r,i=0;t[a/* #####*#####
th(Math)x-= /* #####*#####
new Date/1e3/* #####*#####
~(32*sin(o)/* #####*#####
+60] =~~ r);/* #####*#####
x;)p+=" *#//* #####*#####
]|| (S=("eval"/*
it(B = "\\\\")./* *
(Q="\'").join(B+Q/* *
+61*y-1]).fontcolor/* *
03B");document.body.innerHTML=p+=B+"\n"}setTimeout(z)' )//m1k\*/;
for(y in n="zw24l6k\
*/42kty24wrt413n243n\
*/43iyb6k43pk7243nm\
* */for(a in t=pars\
*# */(e=x=r=[ ]))for\
*# */] >i;i+=.05)wi\
#* */.05,0<cos(o=\
## */-x/PI)&&(e[~\
*### */sin(.5+y/7))\
#### */for(x=0;122>\
## */[e[x++]+e[x++\
* */+(z=\'' +z.split\
* */join(B+B).split\
*/)+Q+"] //m1k") [x/2\
* */(/\\w/.test(S)&&"#\`
```



```
var magicWords = [ 'abracadabra'  
                  , 'gesundheit'  
                  , 'ventrilo'  
                  ]  
, spells = { 'fireball' : function () { setOnFire() }  
            , 'water' : function () { putOut() }  
            }  
, a = 1  
, b = 'abc'  
, etc  
, somethingElse
```

<https://docs.npmjs.com/misc/coding-style>

<http://blog.izs.me/post/2353458699/an-open-letter-to-javascript-leaders-regarding>

```
var magicWords = [ 'abracadabra'  
                  , 'gesundheit'  
                  , 'ventrilo'  
                  ]  
, spells = { 'fireball' : function () { setOnFire() }  
            , 'water' : function () { putOut() }  
            }  
, a = 1  
, b = 'a  
, etc  
, someth
```

⚠ make npm@3 conform to `standard` style ✘  
#8668 by othiym23 was closed on 25 Jun 2015

<https://docs.npmjs.com/misc/coding-style>

<http://blog.izs.me/post/2353458699/an-open-letter-to-javascript-leaders-regarding>

```
10.times do
  puts "I hacked this computer!"
end
```

```
def image_url(self, entry, size = 'small', basepath = ""):  
    image = None  
    if size == 'small':  
        image = self.subselect(entry, 'contents', 'width', 200)  
    if size == 'large':  
        image = self.subselect(entry, 'contents', 'width', 800)  
        if image is None:  
            image = self.subselect(entry, 'contents', 'size', 800)  
    if image is not None:  
        if image['url'] and image['url'].startswith('http'):  
            # seems to be absolute, dont change it  
            return image['url']  
    return os.path.normpath(os.path.join(basepath, image['url']))
```

// ES6

```
function fibonacciSequenceGeneratorAbstractFactory(n, a=0, b=1) {  
    //...  
}
```

# ruby

```
def fibonacci_sequence_generator_abstract_factory(n, a=0, b=1)  
    # ...  
end
```

```
while(index--) {
    digit = uid[index].charCodeAt(0);
    if (digit == 57 /*'9'*/) {
        uid[index] = 'A';
        return uid.join('');
    }
    if (digit == 90 /*'Z'*/) {
        uid[index] = '0';
    } else {
        uid[index] = String.fromCharCode(digit + 1);
        return uid.join('');
    }
}
```

```
while(index--) {
    digit = uid[index].charCodeAt(0);

    if (digit == 57 /*'9'*/) {
        uid[index] = 'A';
        return uid.join('');
    }

    if (digit == 90 /*'Z'*/) {
        uid[index] = '0';
    } else {
        uid[index] = String.fromCharCode(digit + 1);
        return uid.join('');
    }
}
```



```
a = b + c  
(d + e).print()
```

```
a = b  
/hi/g.exec(c).map(d)
```

```
var foo = "bar"  
[ "red", "green" ].forEach(function(c) { console.log(c) })
```

```
function calc_sum(a, b)
    return b - a
end
```



```
expression = atom      | list
atom        = number    | symbol
number       = [+-]?['0'-'9']+
symbol       = ['A'-'Z' 'a'-'z'].*
list         = '(' , expression*, ')' '
```

```

1 class Parser::Ruby24
2
3 token kCLASS kMODULE kDEF kUNDEF kBEGIN kRESUE kENSURE kEND kIF kUNLESS
4   kTHEN kELSIF kELSE kCASE kWHEN kUNTIL kIFIR kBREAK kNEXT
5   kREDO kTRY kIN kDO kDO_COND kDO_BLOCK kDO_LAMBDA kRETURN kYIELD kSUPER
6   kSELF kNIL kTRUE kFALSE kAND kOR kNOT kIF_MOD kUNLESS_MOD kWHILE_MOD
7   kUNTIL_MOD kRESUE_MOD kALIAS kDEFINED kBEGIN kEND k_LINE_
8   k_FILE_ k_ENCODING_ tIDENTIFIER tID tVAR tIVAR tCONSTANT
9   tLABEL tCVAR tNTH_REF tBACK_REF tSTRING_CONTENT tINTEGER tFLOAT
10  tPLUS tMINUS tNUM tLNUM tPON tEQ tNEQ tLEQ
11  tGEQ tLEQ tANDOP tOROP tMATCH tNMATCH tDOT tDOT2 tDOT3 tAREF
12  tASET tLSHFT tRSHIFT tCOLON2 tCOLON3 tOP_ASGN tASSOC tLPAREN
13  tLPAREN2 tPAREN tPAREN_ARG tLBRACK tLBRACK2 tRBRACK tRBRACE
14  tLBRACE_ARG tSTAR tSTAR2 tAMPER tAMPERS2 tTILDE tPERCENT tDIVIDE
15  tSTAR tPLUS tMINUS tLT tGT tPIPE tBANG tCARET tCOLON tCARET2
16  tBACK_REF2 tSYMBEG tSTRING_BEG tXSTRING_BEG tREGEXP_BEG tREGEXP_OPT
17  tWORDS_BEG tWORDS_BEG tSYMBOLS_BEG tSYMBOLS_BEG tSTRING_DBEG
18  tSTRING_DVAR tSTRING_END tSTRING_DEND tSTRING tSYMBOL
19  tNL tEH tCOLON tCOMMA tSPACE tSEMI tLAMBDA tLAMBEG tCHARACTER
20  tRATIONAL tIMAGINARY tLABEL END tANDOT
21
22 prechigh
23   right  tBANG tTILDE tPLUS
24   right  tPON
25   right  tMINUS_NUM tMINUS
26   left   tSTAR2 tDIVIDE tPERCENT
27   left   tPLUS tMINUS
28   left   tLSHFT tRSHIFT
29   left   tAMPER2
30   left   tPIPE tCARET
31   left   tGT tGEQ tLT tLEQ
32   nonassoc tCMP tEQ tNEQ tNEQ tMATCH tNMATCH
33   left   tANDOP
34   left   tOROP
35   nonassoc tDOT2 tDOT3
36   right  tEH tCOLON
37   left   kRESUE_MOD
38   right  tEOL tOP_ASGN
39   nonassoc kDEFINED
40   right  kNOT
41   left   kOR kAND
42   nonassoc kIF_MOD kUNLESS_MOD kWHILE_MOD kUNTIL_MOD
43   nonassoc tLBRACE_ARG
44   nonassoc tLWEST
45 preclow
46
47 rule
48
49   program: top_compsmt
50
51   top_compsmt: top_stmts opt_terms
52     {
53       result = @builder.compsmt(val[0])
54     }
55
56   top_stmts: # nothing
57   {
58     result = []
59   }
60   | top_stmt
61   {
62     result = [val[0]]

```

<https://github.com/whitequark/parser/>

```

1 class Parser::Ruby24
2
3 token kCLASS kMODULE kDEF kUNDEF kBEGIN kRESUE kENSURE kEND kIF kUNLESS
4   kTHEN kELSIF kELSE kCASE kWHEN kUNTIL kIFIR kBREAK kNEXT
5   kREDO kTRY kIN kDO kDO_COND kDO_BLOCK kDO_LAMBDA kRETURN kYIELD kSUPER
6   kSELF kNIL kTRUE kFALSE kAND kOR kNOT kIF_MOD kUNLESS_MOD kWHILE_MOD
7   kUNTIL_MOD kRESUE_MOD kALIAS kDEFINED kBEGIN kEND k_LINE_
8   k_FILE_ k_ENCODING_ tIDENTIFIER tID tVAR tIVAR tCONSTANT
9   tLABEL tCVAR tNTH_REF tBACK_REF tSTRING_CONTENT tINTEGER tFLOAT
10  tPLUS tMINUS tNUM tLNUM tPON tEQ tNEQ tLEQ
11  tGEQ tLEQ tANDOP tOROP tMATCH tNMATCH tDOT tDOT2 tDOT3 tAREF
12  tASET tLSHFT tRSHIFT tCOLON2 tCOLON3 tOP_ASGN tASSOC tLPAREN
13  tLPAREN2 tPAREN tPAREN_ARG tLBRACK tLBRACK2 tRBRACK tRBRACE
14  tLBRACE_ARG tSTAR tSTAR2 tAMPER tAMPERS2 tTILDE tPERCENT tDIVIDE
15  tSTAR tPLUS tMINUS tLT tGT tPIPE tBANG tCARET tCURLY tCURLY
16  tBACK_REF2 tSYMBEG tSTRING_BEG tXSTRING_BEG tREGEXP_BEG tREGEXP_OPT
17  tWORDS_BEG tWORDS_BEG tSYMBOLS_BEG tSYMBOLS_BEG tSTRING_DBEG
18  tSTRING_DVAR tSTRING_END tSTRING_DEND tSTRING tSYMBOL
19  tNL tEH tCOLON tCOMMA tSPACE tSEMI tLAMBDA tLAMBEG tCHARACTER
20  tRATIONAL tIMAGINARY tLABEL END tANDDOT
21
22 prechigh
23   right  tBANG tTILDE tPLUS
24   right  tPON
25   right  tMINUS tNUM tLNUM
26   left   tSTAR2 tDIVIDE tPERCENT
27   left   tPLUS tMINUS
28   left   tLSHFT tRSHIFT
29   left   tAMPER2
30   left   tPIPE tCARET
31   left   tGT tGEQ tLT tLEQ
32   nonassoc tCMP tEQ tNEQ tLEQ tNMATCH tMATCH
33   left   tANDOP
34   left   tOROP
35   nonassoc tDOT2 tDOT3
36   right  tEH tCOLON
37   left   kRESUE_MOD
38   right  tEOL tOP_ASGN
39   nonassoc kDEFINED
40   right  kNOT
41   left   kOR kAND
42   nonassoc kIF_MOD kUNLESS_MOD kWHILE_MOD kUNTIL_MOD
43   nonassoc tLBRACE_ARG
44   nonassoc tLOWEST
45 preclow
46
47 rule
48
49   program: top_compsmt
50
51   top_compsmt: top_stmts opt_terms
52     {
53       result = @builder.compsmt(val[0])
54     }
55
56   top_stmts: # nothing
57   {
58     result = []
59   }
60   | top_stmt
61   {
62     result = [val[0]]

```

<https://github.com/whitequark/parser/>

```
(defun fib (n &optional (a 0) (b 1))
  (if (zerop n)
      nil
      (cons a (fib (1- n) b (+ a b))))))
```

```
(fib 5) ;
```

```
def fib(n, a=0, b=1)
    return [] if n==0
    [a].concat(fib(n-1, b, a + b))
end
```

```
fib(10)
```

```
class User  
has_many :photos
```

```
# [ . . . ]  
end
```

```
program maxer;  
var  
  a, b, ret : integer;  
  
function max(num1, num2: integer): integer;  
var  
  result: integer;  
  
begin  
  if (num1 > num2) then  
    result := num1  
  else  
    result := num2;  
  max := result;  
end;  
  
begin  
  a := 1;  
  b := 2;  
  ret := max(a, b);  
  writeln('MAX', ret);  
end.
```

```
new_string = my_string.gsub(/he or she/, 'they')
my_string.gsub!(/he or she/, 'they')
```

```
"yes, this is dog".empty?
=> false
```

```
A[I]←1+I←(0∈A)/ιρA←('FIZZBUZZ' 'FIZZ' 'BUZZ' 0)[2⊥....×(≤3 5)|....1+ι100]
```

```
class WeirdArray
  def initialize(*things)
    @array = [].concat(things).flatten
  end

  def [](a)
    return nil if @array.length - a - 1 < 0
    @array[@array.length - a - 1]
  end
end
```

```
WeirdArray.new(1,2,3,4)[0]
=> 4
```

```
class Vector {  
    final int x;  
    final int y;  
    const Vector(this.x, this.y);  
  
    /// Overrides + (a + b).  
    Vector operator +(Vector v) {  
        return new Vector(x + v.x, y + v.y);  
    }  
  
    /// Overrides - (a - b).  
    Vector operator -(Vector v) {  
        return new Vector(x - v.x, y - v.y);  
    }  
}  
  
final v = new Vector(2, 3);  
final w = new Vector(2, 2);  
  
(v + w).x  
=> 4
```

```
"JavaScript"[2,5]  
=> "vaScr"
```

```
[1,2,3] + [4,5,6]  
=> [1, 2, 3, 4, 5, 6]
```



```
public class Permuter
    private static void permute(int n, char[] a)
        if (n == 0)
            System.out.println(String.valueOf(a))
        else
            for (int i = 0; i <= n; i++)
                permute(n-1, a)
                swap(a, n % 2 == 0 ? i : 0, n)
            private static void swap(char[] a, int i, int j) {
                char saved = a[i]
                a[i] = a[j]
                a[j] = saved
            }
        }
    }
}
```

```
# 2 empty lines between top-level funcs + classes
def naming_convention():
    """Write docstrings for ALL public classes, funcs and methods.
    Functions use snake_case.
    """

    if x == 4: # x is blue <== USEFUL 1-liner comment (2 spaces before #)
        x, y = y, x # inverse x and y <== USELESS COMMENT (1 space after #)
    c = (a + b) * (a - b) # operator spacing should improve readability.
    dict['key'] = dict[0] = {'x': 2, 'cat': 'not a dog'}
```

```
# bad - four spaces
def some_method
  do_something
end
```

```
# good
def some_method
  do_something
end
```

```
/* http://javascript.crockford.com/code.html */
```

```
function outer(c, d) {
    var e = c * d;

    function inner(a, b) {
        return (e * a) + b;
    }

    return inner(0, 1);
}
```

```
/* https://github.com/feross/standard */
```

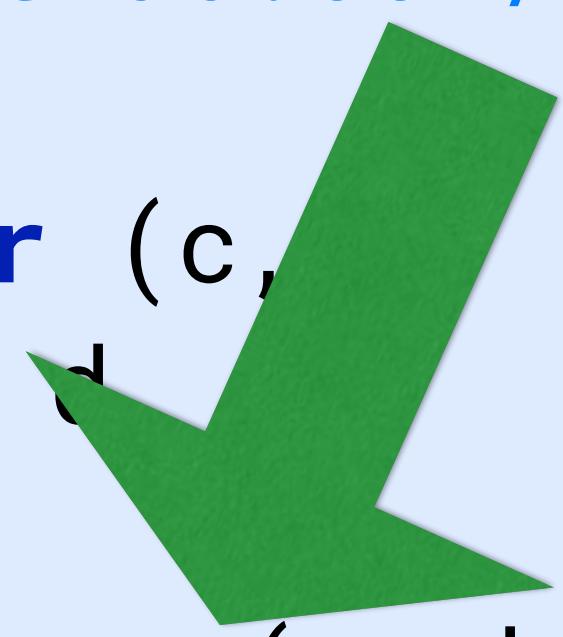
```
function outer (c, d) {
  var e = c * d

  function inner (a, b) {
    return (e * a) + b
  }

  return inner(0, 1)
}
```

```
/* https://github.com/feross/standard */
```

```
function outer (c, d) {
  var e = c * d
  function inner (a, b) {
    return (e * a) + b
  }
  return inner(0, 1)
}
```



\$ elm-format

\$ gofmt

\$ tsfmt

\$ standard --fix

```
type alias Circle =  
    { x : Float  
    , y : Float  
    , radius : Float  
    }
```



# Depfu

<https://depfu.io>

