

Cool Code

@KevlinHenney

cool, *adjective*

- fashionably attractive or impressive
- excellent
- used to express acceptance or agreement
- used as an intensive
- used when a conversation goes silent
- marked by deliberate effrontery or lack of due respect or discretion
- restrained or relaxed in style

code, *noun*

- a system of words, figures or symbols used to represent others
- a set of instructions for a computer
- a computer program, or a portion thereof
- a set of conventions or principles governing behaviour or activity in a particular domain
- a system or collection of rules or regulations on any subject

STR
16571 LD C A 79
(187) LD L A 195
(64) LD H A 38 67
16575 LD B (HL) 126
(131) LD B N 6 1
(64) AND N 230 127
CP N 254 8
JP Z DIS 40 28
INC B 4
CP N 254 110
JPZDIS 40 15
CP N 254 39
JP C DIS 56 11
LD A (HL) 126 '0' DIFF
INC B 4
LD L N 46 25 '1' EMPTY
ADD (HL) 134
BIT 7, A 289 127 '2' WALL
JP Z DIS 40 2
LD B N 6 0 '3' SAME
LD A B 120
LD L C 195
RET 201

TABLES
16607 1 11 -1 -11 -10 -12 10 10
16615 13 -13 21 -21 23 -23 -9 9
16623 11 10 12
16626 34 55 39 51 53
PIECE
16631 XOR A 175
(247) LD (NN) A 50 70 64
(64) LD A (HL) 126
AND N 230 127
CP N 254 53
JP Z DIS 40 79
LD C N 14 1
LD B N 6 0
LD HL NN 33 231 64
CP N 254 51
JP Z DIS 40 22
LD L N 46 223
CP N 254 48
JP Z DIS 40 16
LD C B 72
CP N 254 54
JP Z DIS 40 11
LD B N 6 4
CP N 254 55
JP Z DIS 40 5
LD L N 46 227
CP N 254 39
RET NZ 192

SHIFT
16882 LD HL NN 33 99 64
(242) LD DE NN 17 70 64
(65) LD BC NN 1 28 0
JP C DIS 56 1
EX DE HL 235
LDIR 237 176
RET 201
PSC
17162 AND N 230 127
(10) LD HL NN 33 242 64
(67) LD B N 6 5
CP (HL) 190
RET Z 200
INC HL 35
DJNZ DIS 16 251
LD A B 120
RET 201

NPSCAN
17046 XOR A 175
(150) LD (NN) A 50 65 64
(66) LD B N 6 86
LD HL NN 33 62 67
INC HL 35
PUSH HL 229
PUSH BC 197
LD E L 93
CALL STR2 205 191 64
CP N 254 3
JP NZ DIS 32 41
LD L E 107
LD (NN) HL 34 7 64
CALL MOVE 205 247 64
CALL TL 205 130 66
JP Z DIS 40 29
LD E A 95
LD D N 22 67
CALL MOVE 205 255 66
EXX 217
AND A 167

CALL SHIFT 205 242 65
CALL CHK 205 1 66
EXX 217
LD (HL) B 112
LD A C 121
LD (DE) A 18
JP C DIS 56 3
CALL SCORE 205 153 65
SCF 55
CALL SHIFT 205 242 65
JP DIS 24 222
POP BC 193
POP HL 225
DJNZ DIS 16 200
LD A (NN) 58 65 64
CP N 254 0
JP Z DIS 40 254 *
LD HL NN 33 69 64
LD A (HL) 126
DEC HL 43
LD E (HL) 94
LD D N 22 67
LD (DE) A 18
DEC HL 43
LD L (HL) 110
LD H D 98
BIT 0, L 209 69
(235) LD (HL) N 54 0
(66) JP Z DIS 40 2
LD (HL) N 54 120
CALL CHGMV 205 247 66
RET 201

INC
17176 LD A L 125
(24) EXX 217
(67) LD (NN) A 50 128 64
CALL SQ.AT 205 16 66
EXX 217
LD A C 121
RET 201

DRIVER
16909 LD B H 6 5
(63) LD A H 62 8
(66) LD HL NN 33 159 67
INC HL 35
LD (HL) A 119
DJNZ DIS 16 252
CALL KT 205 168 64
CP N 254 3
JP NZ DIS 32 238
LD (NN) HL 34 7 64
LD E L 93
CALL MOVE 205 247 64
LD HL NN 33 161 67
CALL KT 205 168 64
CP N 245 2
EX DE HL 235
JP NC DIS 48 220
CALL TL 205 130 66
JP Z DIS 40 215
CP C 185
JP NZ DIS 32 248
CALL MOVE 205 255 66
EXX 217
CALL CHK 205 1 66
EXX 217
JP C DIS 56 8
CALL CHG SQ 205 235 66
CALL NPSCAN 205 150 66
JP DIS 24 194
LD (HL) B 112
LD A C 121
LD (DE) A 18
JP DIS 24 249

TEST LIST
17026 LD HL NN 33 70 64
(130) DEC (HL) 53
(66) LD A (HL) 126
INC A 68
RET Z 200
ADD L 133
LD L A 111
LD A (HL) 126
RET 201

PANN
16721
(81)
(65)

LD A (HL) 126
AND N 230 128
LD HL NN 33 228 64
JP NZ DIS 32 2
LD L N 46 241
LD D N 22 3
LD A E 123
ADD (HL) 134
PUSH HL 229
PUSH AF 245
CP N 254 63
JP C DIS 56 32
CP N 254 148
JP NC DIS 48 28
CALL STR 205 187 64
CP N 254 0
JP Z DIS 40 20
CP N 254 1
JP NZ DIS 32 17
LD A D 122
CP N 254 1
JP NZ DIS 32 12
CALL ALIST 205 141 66
LD A E 123
CP N 254 82
JP C DIS 56 19
CP N 254 126
JP NC DIS 48 15
POP AF 241
POP HL 225
DEC INC HL 43 !
DEC D 21
JP NZ DIS 32 210
RET 201
LD A D 122
CP N 254 1
CALL NZ ALIST 196 141 66
JP DIS 24 241
POP AF 241
POP HL 225
LD E A 95
JP DIS 24 197

DEC
16897 LD A (NN) 58 55 67
(1) ADD N 198 48
(66) LD HL NN 33 62 67
LD B A 71
CDIR 237 177
DEC HL 43
LD (NN) HL 34 120 64
LD B N 6 86
LD HL NN 33 62 67
INC HL 35
PUSH HL 229
PUSH BC 197
LD E L 93
CALL STR2 205 191 64
CP 0 254 0
JP NZ DIS 32 25
CALL CHEMV 205 247 66
LD L E 107
CALL MOVE 205 247 64
CALL CHEMV 205 247 66
CALL TL 205 130 66
JP Z DIS 40 10
LD HL (NN) 42 128 64

CHK
16997 LD A (NN) 58 55 67
(1) ADD N 198 48
(66) LD HL NN 33 62 67
LD B A 71
CDIR 237 177
DEC HL 43
LD (NN) HL 34 120 64
LD B N 6 86
LD HL NN 33 62 67
INC HL 35
PUSH HL 229
PUSH BC 197
LD E L 93
CALL STR2 205 191 64
CP 0 254 0
JP NZ DIS 32 25
CALL CHEMV 205 247 66
LD L E 107
CALL MOVE 205 247 64
CALL CHEMV 205 247 66
CALL TL 205 130 66
JP Z DIS 40 10
LD HL (NN) 42 128 64

SQ.AT
16912 LD HL NN 33 62 67
(16) INC HL 35
(66) PUSH HL 229
PUSH BC 197
LD E L 93
CALL STR2 205 191 64
CP 0 254 0
JP NZ DIS 32 25
CALL CHEMV 205 247 66
LD L E 107
CALL MOVE 205 247 64
CALL CHEMV 205 247 66
CALL TL 205 130 66
JP Z DIS 40 10
LD HL (NN) 42 128 64

CP L 189
JP NZ DIS 32 245
POP BC 193
POP HL 225
SCF 55
RET 201
POP BC 193
POP HL 225
DJNZ DIS 16 216
AND A 167
RET 201

SCORE
16793
(153)
(65)
PUSH HL 229
PUSH BC 197
PUSH DE 213
PUSH HL 229
PUSH BC 197
LD D L 85
LD HL NN 33 64 64
CALL NN 205 36 7
CALL PSC 205 10 67
LD A B 120
ADD A N 132
LD C A 79
POP AF 241
CALL PSC 205 10 67
POP HL 225
CALL INC 205 24 67
JP NC DIS 48 1
ADD A B 128
LD C A 79
POP HL 225
POP DE 209
LD E (HL) 94
LD (HL) D 114
PUSH HL 229
PUSH DE 213
CALL INC 205 24 67
JP NC DIS 48 1
SUB B 144
PUSH AF 245
CALL CHEMV 205 247 66
CALL CHK 205 1 66
POP BC 193
JP NC DIS 48 2
INC B 4
INC B 4
POP DE 209
POP HL 225
LD (HL) E 115
POP HL 225
CALL CHG 205 250 66
CALL INC 205 24 67
JP NC DIS 48 1
DEC B 5
CALL CHG 205 250 66
CALL CHEMV 205 247 66
LD A B 120
LD HL NN 33 60 64
LD (HL) A 119
EX DE HL 235
LD HL NN 33 65 64
CP (HL) 190
RET C 216
LD BC NN 1 5 0
JP DIS 24 11

HERE IS THE PHILOSOPHY OF GULDENSTERN: ON EVERY APPEARANCE OR DISAPPEARANCE OF THE MANUAL THROTTLE
 # DISCRETE TO SELECT P67 OR P66 RESPECTIVELY: ON EVERY APPEARANCE OF THE ATTITUDE-HOLD DISCRETE TO SELECT P66
 # UNLESS THE CURRENT PROGRAM IS P67 IN WHICH CASE THERE IS NO CHANGE

GULDEN	EXTEND		# IS UN-AUTO-THROTTLE DISCRETE PRESENT?
# STERN			# RSB 2009: Not originally a comment.
	READ CHAN30		
	MASK BIT5		
	CCS A		
P67NOW?	TCF STARTP67		# YES
	TC CHECKMM		# NO: ARE WE IN P67 NOW?
	DEC 67		
STARTP66	TCF STABL?		# NO
	TC FASTCHNG		# YES
	TC NEWMODEX		
DEC66	DEC 66		
	EXTEND		
	DCA HDOTDISP		# SET DESIRED ALTITUDE RATE - CURRENT
	DXCH VDGVERT		# ALTITUDE RATE.
STRTP66A	TC INTPRET		
	SLOAD PUSH		
	SLOAD PBIAS		
	SLOAD PBIAS		
	SLOAD VDEF		
	SLOAD PBIASK		
	VXSC SET		
	STOVL BIASFACT		
	STOVL RODFLAG		
	STOVL VBIAS		
	STOVL TEMX		
	VCOMP		
	STOVL OLDPIPAK		
	STOVL ZEROVECS		
	STODL DELVRD		
	STODL RODSCALE		
	STODL RODSCALE1		
	STODL PIPTIME		
	STORE LASTTPIP		
	EXIT		
	CAF ZERO		
	TS FCOLD		
	TS FWEIGHT		
	TS FWEIGHT +1		
	TS WCHVERT		
VRTSTART			
# Page 801			
	CAF TWO		# WCHPHASE - 2 ----> VERTICAL: P65,P66,P67
	TS WCHPHOLD		
	TS WCHPHASE		
	TC BANKCALL		# TEMPORARY, I HOPE HOPE HOPE
	CADR STOPRATE		# TEMPORARY, I HOPE HOPE HOPE
	TC DOWNFLAG		# PERMIT X-AXIS OVERRIDE
	ADRES XOVINFLG		
	TC DOWNFLAG		
	ADRES REDFLAG		
	TCF VERTGUID		
STARTP67	TC NEWMODEX		# NO HARM IN "STARTING" P67 OVER AND OVER
	DEC 67		# SO NO NEED FOR A FASTCHNG AND NO NEED
	CAF ZERO		# TO SEE IF ALREADY IN P67.
	TS RODCOUNT		
	CAF TEN		
	TCF VRTSTART		
STABL?	CAF BIT13		# IS UN-ATTITUDE-HOLD DISCRETE PRESENT?
	EXTEND		
	RAND CHAN31		
	CCS A		
	TCF GULDRET		# YES ALL'S WELL
P66NOW?	CS MODREG		
	AD DEC66		
	EXTEND		
	BZF RESTART?		
	CA RODCOUNT		# NO. HAS THE ROD SWITCH BEEN "CLICKED"?
	EXTEND		
	BZF GULDRET		# NO. CONTINUE WITH AUTOMATIC LANDING
	TCF STARTP66		# YES. SWITCH INTO THE ROD MODE.
RESTART?	CA FLAGWRD1		# HAS THERE BEEN A RESTART?
	MASK RODFLBIT		
	EXTEND		
	BZF STRTP66A		# YES. REINITIALIZE BUT LEAVE VDGVERT AS
			# IS.
	TCF VERTGUID		# NO: CONTINUE WITH R.O.D.

```

/* grep: search for regexp in file */
int grep(char *regexp, FILE *f, char *name)
{
    int n, nmatch;
    char buf[BUFSIZ];

    nmatch = 0;
    while (fgets(buf, sizeof buf, f) != NULL) {
        n = strlen(buf);
        if (n > 0 && buf[n-1] == '\n')
            buf[n-1] = '\0';
        if (match(regexp, buf)) {
            nmatch++;
            if (name != NULL)
                printf("%s:", name);
            printf("%s\n", buf);
        }
    }
    return nmatch;
}

/* matchhere: search for regexp at beginning of text */
int matchhere(char *regexp, char *text)
{
    if (regexp[0] == '\0')
        return 1;
    if (regexp[1] == '*')
        return matchstar(regexp[0], regexp+2, text);
    if (regexp[0] == '$' && regexp[1] == '\0')
        return *text == '\0';
    if (*text != '\0' && (regexp[0] == '.' || regexp[0] == *text))
        return matchhere(regexp+1, text+1);
    return 0;
}

/* match: search for regexp anywhere in text */
int match(char *regexp, char *text)
{
    if (regexp[0] == '^')
        return matchhere(regexp+1, text);
    do { /* must look even if string is empty */
        if (matchhere(regexp, text))
            return 1;
    } while (*text++ != '\0');
    return 0;
}

/* matchstar: search for c*regexp at beginning of text */
int matchstar(int c, char *regexp, char *text)
{
    do { /* a * matches zero or more instances */
        if (matchhere(regexp, text))
            return 1;
    } while (*text != '\0' && (*text++ == c || c == '.'));
    return 0;
}

```

```

/**
 * Runs the bare test sequence.
 * @exception Throwable if any exception is thrown
 */
public void runBare() throws Throwable {
    setUp();
    try {
        runTest();
    }
    finally {
        tearDown();
    }
}

/**
 * Override to run the test and assert its state.
 * @exception Throwable if any exception is thrown
 */
protected void runTest() throws Throwable {
    Method runMethod= null;
    try {
        // use getMethod to get all public inherited
        // methods. getDeclaredMethods returns all
        // methods of this class but excludes the
        // inherited ones.
        runMethod= getClass().getMethod(fName, null);
    } catch (NoSuchMethodException e) {
        fail("Method \""+fName+"\" not found");
    }
    if (!Modifier.isPublic(runMethod.getModifiers())) {
        fail("Method \""+fName+"\" should be public");
    }

    try {
        runMethod.invoke(this, new Class[0]);
    }
    catch (InvocationTargetException e) {
        e.fillInStackTrace();
        throw e.getTargetException();
    }
    catch (IllegalAccessException e) {
        e.fillInStackTrace();
        throw e;
    }
}
}

```

```

import re, collections

def words(text): return re.findall('[a-z]+', text.lower())

def train(features):
    model = collections.defaultdict(lambda: 1)
    for f in features:
        model[f] += 1
    return model

NWORDS = train(words(file('big.txt').read()))

alphabet = 'abcdefghijklmnopqrstuvwxyz'

def edits1(word):
    splits      = [(word[:i], word[i:]) for i in range(len(word) + 1)]
    deletes     = [a + b[1:] for a, b in splits if b]
    transposes  = [a + b[1] + b[0] + b[2:] for a, b in splits if len(b)>1]
    replaces    = [a + c + b[1:] for a, b in splits for c in alphabet if b]
    inserts     = [a + c + b      for a, b in splits for c in alphabet]
    return set(deletes + transposes + replaces + inserts)

def known_edits2(word):
    return set(e2 for e1 in edits1(word) for e2 in edits1(e1) if e2 in NWORDS)

def known(words): return set(w for w in words if w in NWORDS)

def correct(word):
    candidates = known([word]) or known(edits1(word)) or known_edits2(word) or [word]
    return max(candidates, key=NWORDS.get)

```

```
(define (eval exp env)
  (cond ((self-evaluating? exp) exp)
        ((variable? exp) (lookup-variable-value exp env))
        ((quoted? exp) (text-of-quotation exp))
        ((assignment? exp) (eval-assignment exp env))
        ((definition? exp) (eval-definition exp env))
        ((if? exp) (eval-if exp env))
        ((lambda? exp)
         (make-procedure (lambda-parameters exp)
                          (lambda-body exp)
                          env))
        ((begin? exp)
         (eval-sequence (begin-actions exp) env))
        ((cond? exp) (eval (cond->if exp) env))
        ((application? exp)
         (apply (eval (operator exp) env)
                 (list-of-values (operands exp) env)))
        (else
         (error "Unknown expression type - EVAL" exp))))
```



```

def eval(x, env=global_env):
    "Evaluate an expression in an environment."
    if isa(x, Symbol):
        # variable reference
        return env.find(x)[x]
    elif not isa(x, list):
        # constant literal
        return x
    elif x[0] == 'quote':
        # (quote exp)
        (_, exp) = x
        return exp
    elif x[0] == 'if':
        # (if test conseq alt)
        (_, test, conseq, alt) = x
        return eval((conseq if eval(test, env) else alt), env)
    elif x[0] == 'set!':
        # (set! var exp)
        (_, var, exp) = x
        env.find(var)[var] = eval(exp, env)
    elif x[0] == 'define':
        # (define var exp)
        (_, var, exp) = x
        env[var] = eval(exp, env)
    elif x[0] == 'lambda':
        # (lambda (var*) exp)
        (_, vars, exp) = x
        return lambda *args: eval(exp, Env(vars, args, env))
    elif x[0] == 'begin':
        # (begin exp*)
        for exp in x[1:]:
            val = eval(exp, env)
        return val
    else:
        # (proc exp*)
        exps = [eval(exp, env) for exp in x]
        proc = exps.pop(0)
        return proc(*exps)

```

```
isa = isinstance
```

```
Symbol = str
```

```

def to_string(exp):
    "Convert a Python object back into a Lisp-readable string."
    return '('+' '.join(map(to_string, exp))+')' if isa(exp, list) else str(exp)

def repl(prompt='lis.py> '):
    "A prompt-read-eval-print loop."
    while True:
        val = eval(parse(raw_input(prompt)))
        if val is not None: print to_string(val)

```

```

v=0000;eval$s=%q~d=%!^Lcf<LK8,          _@7gj*LJ=c5nM)Tplg0%Xv.,S[<>YoP
4ZojjV)O>qIH1/n[|2yE[>:ieC          "%.#%   :::##"   97N-A&Kj_K_><wS5rtWk@*a+Y5
yH?b[F^e7C/56j|pmRe+:)B          "##%   :#####"   O98(Zh)T_Iof*nm.,$C5Nyt=
PPu01Avw^<IiQ=5$'D-y?          "##:   #####"   g6`YT+qLw9k^ch|K'),tc
6ygIL8xI#LNz3v}T=4W          "#   #.   .#####"   lL27FZ0ij)7TQCI)P7u
}RT5-iJbbG5P-DHB<.          "   ##### # :#####"   R,YvZ_rnv6ky-G+4U'
$*are@b4U351Q-ug5          "   #####"   00x8RR%`Om7VDp4M5
PFixrPvl&<p[]lIJ          "   #####:#### %#####"   EGgDt8Lm#;bc4zS^
y]0`PstfUxOC(q          "   .#####:##%   .##   ."   /,}.YOIFj(k&q_V
zcaAī?] ^lCVYp!;          " %%   .#####.   #.   "   ;s="v=%04o;ev"%
(;v=(v-($*+[45,          " :####:   :   "   ]) [n=0].to_i);%
360)+"al$s=%q#{          "%#####.   #####   "   ;;"%c"%126+$s<<
126}";d.gsub!(/          "#####.   #####   "   |\s|".*"/, "");;;
require"zlib"||          "#####   :#####.   "   ;d=d.unpack"C*"
d.map{|c|n=(n||          " :#####:   .#####:   "   ) *90+(c-2)%91};
e=["%x"%n].pack          " :#####%   :##### #:   "   &"H*";e=Zlib::
Inflate.inflate(          " #####%   .#####% ::   "   &&e).unpack("b*"
)[0];22.times{|y|          " #####   %###   "   ;w=(Math.sqrt(1-(
(y*2.0-21)/22)**(          " .###:   .#%   "   ;2)) *23).floor;(w*
2-1).times{|x|u=(e+          " %##   "   ) [y*z=360,z]*2;u=u[
90*x/w+v+90,90/w];s[(          " #.   "   ;y*80)+120-w+x]=(""<<
32<<".:;%#") [4*u.count((          " .   "   ;"0"))/u.size]}};puts\
s+";_The_Globe#{          "#   :#####"   ;"Copyright(C).Yusuke End\
oh, 2010")}" ;exit~;_The_Globe          Copyright(C).Yusuke Endoh, 2010

```

`/^1?$ | ^(11+?) \1+$/`

```

// Erwin Unruh, untitled program,
// ANSI X3J16-94-0075/ISO WG21-462, 1994.

template <int i>
struct D
{
    D(void *);
    operator int();
};

template <int p, int i>
struct is_prime
{
    enum { prim = (p%i) && is_prime<(i>2?p:0), i>::prim };
};

template <int i>
struct Prime_print
{
    Prime_print<i-1> a;
    enum { prim = is_prime<i,i-1>::prim };
    void f() { D<i> d = prim; }
};

struct is_prime<0,0> { enum { prim = 1 }; };
struct is_prime<0,1> { enum { prim = 1 }; };
struct Prime_print<2>
{
    enum { prim = 1 };
    void f() { D<2> d = prim; }
};

void foo()
{
    Prime_print<10> a;
}

// output:
// unruh.cpp 30: conversion from enum to D<2> requested in Prime_print
// unruh.cpp 30: conversion from enum to D<3> requested in Prime_print
// unruh.cpp 30: conversion from enum to D<5> requested in Prime_print
// unruh.cpp 30: conversion from enum to D<7> requested in Prime_print
// unruh.cpp 30: conversion from enum to D<11> requested in Prime_print
// unruh.cpp 30: conversion from enum to D<13> requested in Prime_print
// unruh.cpp 30: conversion from enum to D<17> requested in Prime_print
// unruh.cpp 30: conversion from enum to D<19> requested in Prime_print

```

If you don't have time
to read, you don't
have the time or the
tools to write.

Stephen King

LIFE IS A GAME

Steven “Doc” List

Product Manager, ThoughtWorks Learning™



Who plays online games?

Вторженец из Северного



Who plays online games?

Вторженец из Северного

sniff



Who has certifications?

Who has certifications?

**Who thinks that they're
valuable?**



Outliers

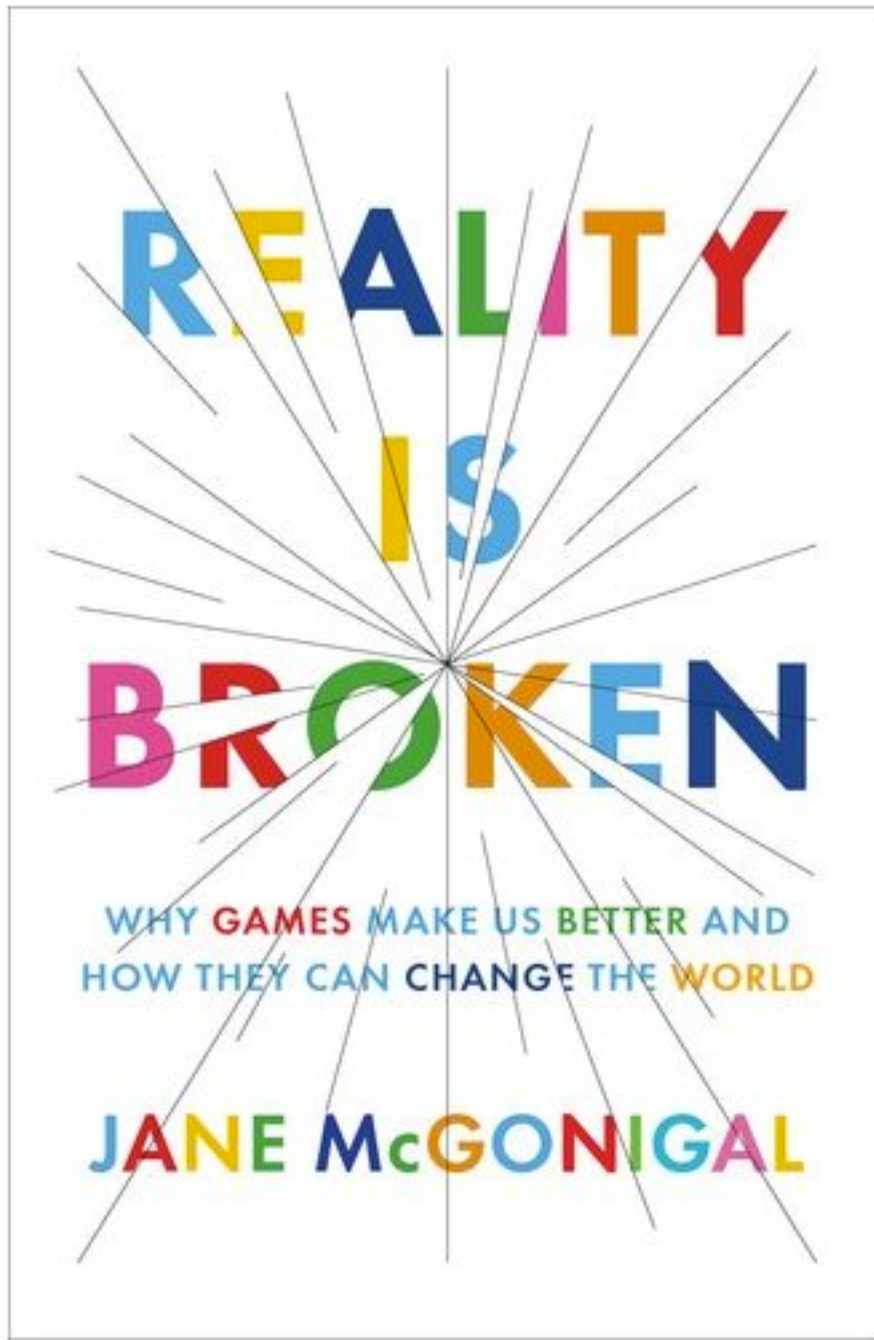


THE STORY OF SUCCESS

MALCOLM
GLADWELL

#1 bestselling author of *The Tipping Point* and *Blink*

Malcolm
Gladwell
“Outliers: The
Story of Success”



Jane McGonigal
“Reality is
Broken:
Why Games Make
Us Better and
How They Can
Change the
World”

Level Up

Level Up

Power Up

Level Up

Power Up

Gain Experience

Level Up

Power Up

Gain Experience

Get Recognition

Level Up

Power Up

Gain Experience

Get Recognition

Collaborate

MASTERY QUEST™

MASTERY QUEST™

IT'S COMING!

Steven “Doc” List

Product Manager, ThoughtWorks Learning™

Doc@ThoughtWorks.com

www.StevenList.com

Mastery-Quest.blogspot.com



THREE

OBJECTIONABLE

THINGS

PETE GOODLIFFE



ACCU 2011, Oxford, UK

@petegoodliffe
pete@goodliffe.net
goodliffe.blogspot.com
www.goodliffe.net



PETE GOODLIFFE
PROGRAMMER / AUTHOR / COLUMNIST / TEACHER ★ CONSCIENTIOUS CODER



```
(public:
//get singleton instance
//static UrbEngine &UrbEngine::GetInstance();

UrbEngine();
virtual ~UrbEngine();

float GetProgramEditQLinkParameter(int which_pad, int qlink_index, char *formattedValueString);
void SetProgramEditQLinkParameter(int which_pad, int qlink_index, float value);
//void getParameterLabel(int index, char *label);
//void getParameterDisplay(int index, char *text);
//void getParameterName(int index, char *text);
void setSampleRate(float sampleRate);
float GetSamplerate() { return _sample_rate; }

void setBlockSize(int blockSize);
int GetBlockSize();
void SetOutputLatencySamples(int latencySamples);

int GetNumMidiInputChannels() { return 16; }
void TriggerMidiCC(int deltaFrames, int index, float value);
void TriggerNote(int deltaFrames, int padKey, bool note_on, float velocity);
void TriggerPad(int deltaFrames, int padKey, bool note_on, float velocity);
void SetPadAftertouchAmount(int pad, int after_touch_amount);
int GetPadAftertouchAmount(int pad);
void SoloInstrument(int which_track, bool solo);

void PlayAuditionSample(const char *filename);
void StopAuditionSample();
int LoadSample(const char *filename, DrumProgram *program, int instrument = 1, int whichlayer = 0);
void SetSampleBuffer(void *wave_buffer, int length, int channels, int wave_sample_rate, vector<float> &slice_info);
void SetupSlicePositionsFromWaveMargin(int which_margin);
void SetFilterCoefficients();
void TriggerSimpleEvent(long note, float velocity);
//void InstrumentNoteOn(Program *program, long delta, SequenceEvent *sequence_event, int base_note);
void InternalNoteOn(Instrument *instrument_ptr, long delta, SequenceEvent *sequence_event, int base_note);
void AddPluginMidiMessage(UrbPluginMidiMessage &pluginMidiMessage);
void GroupNoteOff(int group, long delta, Program *which_program);
void NoteOff(long note, long delta, Program *which_program = 0);
void SetCrunchMode(int crunch_mode);
void SetFactoryContentDirectory(const char *dir);
std::string &GetFactoryContentDirectory();
//static void QuantizeTimeCode(long &time_code, float swing_level, long quantization_value, bool quantize_to_beginning);
void SetSequenceEventValue(int event_index, float value);
void SetSequenceTempo(float tempo);
void CalculateSamplesPerPulse();
vector<SequenceEvent*>::iterator DeleteSequenceEvent(int event_index);
int GetSelectedInstrument();
void SetSelectedInstrument(int instrument);

bool SetInstrumentNameIfSingle(int whichone, const string &name);
string &GetSelectedWaveFileName();
string &GetSelectedWaveFileName(int instrument, int whichlayer);
vector<int> &GetSongPatterns();
void WritePreSetsArray();
vector<int> &GetSongPatternList();
int GetCrunchMode();
void SetSliceSensitivity(float slice_sensitivity);
int GetPadMode();
//int PadControlClicked(int index, float value);
void NewSong();

string &GetPluginDirectory();
unsigned long GetVersionNumber(){return _version_number;};
void DoPatternsSequenceFunction(int value, int selected_pattern);
void DeleteGroup(int which_group);

void SetInstrumentGroup(int which_instrument, float whichGroup);
int GetInstrumentGroup(int which_instrument);

bool IsInstrumentMute(int which_instrument);

//void DeleteDoubleEntries();

//drum program
void SetCurrentProgramName(const char * name);
const char *GetCurrentProgramName();
//const char *GetSelectedFxParameterName(int index);
bool *GetPadTriggered();
bool LoadLoopIntoSequence(const char *path, int sequence);
//bool ImportLoop(const char *filename, bool repeat_or_truncate_old_patterns);
Program *GetCurrentProgram();
void SetProgramType(int type);
void ClearProgram(Program *program, bool clear_associated_patterns);
SamplePool *GetSamplePoolPtr();

int GetNumberOfSlices();
//void SetChannelMode(int channel_mode);
FunctionMapper *GetFunctionMapper(){return &function_mapper;};
ParameterMapper *GetParameterMapper(){return &parameter_mapper;};
LastConfigAction *GetLastMappedAction() {return &last_config_action;};

int GetFunctionTagFromNote(int function) {return function_mapper.GetFunctionTagFromNote(int function);};
void SetFunctionNote(int function_slot, int note, bool learn) {function_mapper.SetNote(function_slot, note, learn);};
int GetParameterFromMidiCC(int controlID) {return parameter_mapper.GetParameterFromMidiCC(controlID);};
void SetParameterMidiCC(int parameter_slot, int midi_cc, bool learn) {parameter_mapper.SetMidiCC(parameter_slot, midi_cc, true);};
//void RecordSample(float **inputs, const long sampleFrames);
Sample *CreateSampleFromRecordingChain();
void ClearRecordingChain();
void Process(float **inputs, float **outputs, long sampleFrames, bool replacing_notes, bool real_time);
void SetCurrentProgram(Program *program);
bool IsExporting() { return _exporting;};
bool ExportSequence(const char *filename, bool is_song, bool export_tracks);
float *GetLeftMainOutMax() {return &left_main_out_max;};
float *GetRightMainOutMax() {return &right_main_out_max;};
float *GetLeftMainInMax() {return &left_main_in_max;};
float *GetRightMainInMax() {return &right_main_in_max;};
Effects *GetEffectsPtr();

void SetSequenceUsed(int sequence, bool used);
bool IsSequenceUsed(int sequence);
bool ExportSequence(const char *filename, bool is_song, bool export_tracks);
bool SaveSongSequence(BlueFile *file);
bool LoadSongSequence(BlueFile *file);
void SetSequenceView(SEQUENCE_VIEW which_view);
SEQUENCE_VIEW GetSequenceView();

//void SetTimeSignature(int beats_per_bar, int beat_length);
int GetCurrentBeatsPerBar();
void SetBeatsPerBar(int beats_per_bar);
inline void ConvertEventTimesAbsoluteToBarOffset();
inline bool AddTracksReturnsNumberofMidiInputChannelsUsedandOldBeatsperbar, int new_beats_per_bar, int old_beats_per_bar, int new_beat_length, int old_beat_length);
void SetSequenceLength(long length);
int GetSequenceLength();
void SetCurrentSequenceLength(long new_track_length);
long GetCurrentSequenceLength();
float GetTempo();

//get singleton instance
//static UrbEngine &UrbEngine::GetInstance();

class Engine
{
public:
//void SetCurrentProgram(Program *program, int track, int whichlayer);
void SetTrackProgram(int track, Program *program);
Track GetTrack(int which_sequence, int track = -1, bool create_if_nonexistent = false);
void SetCurrentTrackIndex(int track);
int GetCurrentTrackIndex();
void SetCurrentTrackMute();
void SoloTrack(int track, bool solo);
void MuteTrack(int track, bool mute);
void CheckForSequenceRunnerBounds();
int GetSequenceNumberOfUsedTracks(INTs which_sequence = -1);
//void SetTrackProgram(int track, Program *program);
void SetTrackProgram(int track, Program *program);
void SetTrackProgram(int track, Program *program);
void SetCurrentTrackProgram(Program *program);
inline void CheckForSequenceRunnerBounds();

//events
inline void ResolveTrackEventIndex(Track *track, int track_index);
inline void ResolveTrackEventIndex(int track_index);
void TriggerSequenceEvents(long sampleFrames);
void CutInstrumentEvents(int instrument to cut = -1);
void CutInstrumentEvents(int instrument to copy = -1);
void RecordSequenceEvent(int deltaFrames, SequenceEvent temp_sequence_event, bool add_command = false, int track = -1);
void SetSequenceEventLength(int deltaFrames, int note, int track = -1);
//void SetAutoQuantize(bool auto_quantize);
//bool IsAutoQuantize();
void SortSequenceEventList();
//void SortSequenceEventList(int &event_index);
long GetQuantizationValue();
void SetQuantizeMode(int mode);
int GetQuantizeMode();
inline long QuantizeTimeCode(long &time_code); //returns the quantized time code before swing
inline long QuantizeTimeCode(long &time_code, float swing_level, long quantization_value, bool quantize_to_beginning);
void Quantize();
void PasteSequenceEventsFromCopyBuffer();
//long GetPPQN();
void MatchPPQNsToSequence();
void SetNoteRepeatActive(bool active);
bool IsNoteRepeatActive();
void SetNoteEraseActive(bool active);
bool IsNoteEraseActive();
Program *CreateNewProgram(PROGRAM_TYPE type);
Program *CreatePluginProgram(const char *file_path, bool setprogram);
void SetPluginProgram (PluginProgram *program);
void SetPluginProgram(const char *plugin_name);
void DeleteDoubleEntries();

//actions
void ActionAddSequenceEvent(SequenceEvent &event_to_add);
void ActionAddSequenceEventToCopyBuffer(SequenceEvent &event_to_add);
void ActionClearCopyBuffer();
long ActionSetAutomationEventPosition(SequenceEvent &sequence_event_to_set, float value, long position);
int ActionSetSequenceEventIndex(SequenceEvent &sequence_event_to_set, int index);
long ActionSetSequenceEventPosition(SequenceEvent &sequence_event_to_set, int note, long position);
long ActionSetSequenceEventTimeCode(SequenceEvent &sequence_event_to_set, long position);
long ActionSetSequenceEventLength(SequenceEvent &sequence_event_to_set, long length);
long ActionSetSequenceEventValue(SequenceEvent &sequence_event_to_set, long time_code, float value);
void ActionDeleteSequenceEvent(SequenceEvent &event_to_delete);

//actions that require true;
void ActionAddTimeSignature(int bar_position, int numerator, int denominator);
void ActionMoveTimeSignature(int bar_position, int new_bar_position);
void ActionChangeTimeSignature(int bar_position, int numerator, int denominator);
void ActionDeleteTimeSignature(int bar_position);
bool GetTimeSignatureNumeratorAndDenominator(int bar_position, int &numerator, int &denominator);

void AddNoteOnEvent(int which_sequence, int which_track, int channel, long time, int note, int velocity);
void AddNoteOffEvent(int which_sequence, int which_track, int bars, int channel, long time, int note, int velocity);
```


3

why?

GOODLIFFE // PAY ATTENTION!



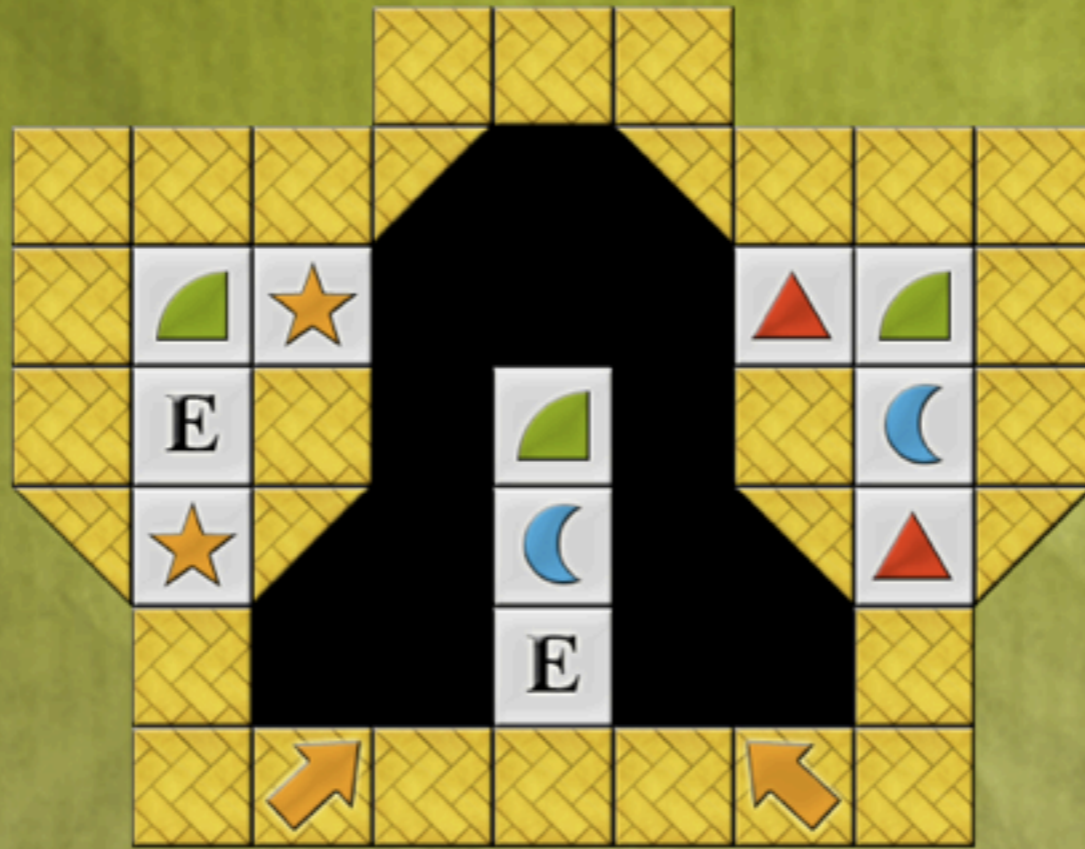
15 minutes!

ASPIDISTRA #1





25. THROWN UP



0:13

0 MOVES



 **Play theme tune**
(AVAudioPlayer)

 **Fade in/out**

Objective-C

Objective-C

categories

associative references

blocks

ROMAN TRAVELOGUE #2



PROBLEM#1



a nice the API

categories

(because one interface is never enough)

```
AVAudioPlayer *player = ... ;  
[player play];  
[player stop];
```

```
AVAudioPlayer *player = ... ;  
[player play];  
[player stop];
```

```
[player playWithFadeDuration:1.0];  
[player stopWithFadeDuration:1.0];
```

```
@interface AVAudioPlayer
```

```
{
```

```
    ...
```

```
}
```

```
- (id) initWithContentsOfURL: (NSURL*)url;
```

```
- (void) play;
```

```
- (void) stop;
```

```
@end
```

```
@interface AVAudioPlayer
```

```
{
```

```
    ...
```

```
}
```

```
- (id) initWithContentsOfURL: (NSURL*)url;
```

```
- (void) play;
```

```
- (void) stop;
```

```
- (void) playWithFadeDuration: (float)secs;
```

```
- (void) stopWithFadeDuration: (float)secs;
```

```
@end
```

```
@interface AVAudioPlayer (Fades)
```

```
- (void) playWithFadeDuration: (float) secs;  
- (void) stopWithFadeDuration: (float) secs;
```

```
@end
```



```
@implementation AVAudioPlayer (Fades)
```

```
- (void) playWithFadeDuration: (float)secs
```

```
{
```

```
    // magic happens here
```

```
}
```

```
- (void) stopWithFadeDuration: (float)secs
```

```
{
```

```
    // clever stuff in here
```

```
}
```

```
@end
```

```
AVAudioPlayer *player = ... ;  
[player play];  
[player stop];
```

```
[player playWithFadeDuration:1.0];  
[player stopWithFadeDuration:1.0];
```



**THIS IS NOT A
SUBLIMINAL
MESSAGE**

EQUIDISTANT CAMOMILE #4



PROBLEM#2



*we need some new
instance variables*

associative references
(a posh name for cheating)



```
static const char volumeLevelKey = 'Q';  
NSNumber number = [NSNumber numberWithInt:1.0];
```

```
objc_setAssociatedObject(self,  
    &volumeLevelKey,  
    &variable,  
    OBJ_ASSOCIATION_RETAIN_NONATOMIC);
```

```
NSNumber *number =  
    (NSNumber*)objc_getAssociatedObject(self, &volumeLevelKey);
```



```
@interface AVAudioPlayer (Fades)
```

```
- (void) playWithFadeDuration: (float)secs;  
- (void) stopWithFadeDuration: (float)secs;
```

```
@property float fadeVolume;
```

```
@end
```

```
- (void) fadeVolume  
{  
    // gibberish in here  
}
```

```
- (void) setFadeVolume  
{  
    // cobblers in here  
}
```

```
float fadeVolume = player.fadeVolume;
```



A TRAVESTY OF PYLON #6



PROBLEM#3



*we need to use another
fancy language feature*

blocks

(because C++ isn't the only cool language)

PROBLEM#3



*when a fade has
completed, do
“something”*

```
typedef void (^FadeCompleteBlock)();
```

```
typedef void (^FadeCompleteBlock)();
```

```
- (void) fadeToVolume:(float)volume  
    withDuration:(float)secs  
    andThen:(FadeCompleteBlock)action
```



```
typedef void (^FadeCompleteBlock)();
```

```
- (void) fadeToVolume:(float)volume  
    withDuration:(float)secs  
    andThen:(FadeCompleteBlock)action
```

```
[player fadeToVolume:0.0  
    withDuration:1.0  
    andThen:^(  
        [player stop];  
        player.volume = 1.0;  
    )];
```

TRANSCENDENTAL SHOE #7



the morale of the story
(there is none)



Pete Goodliffe

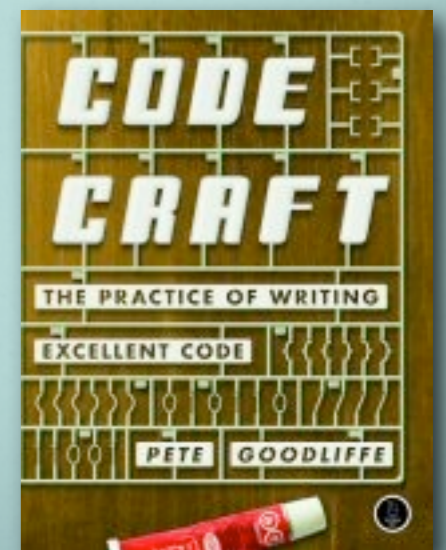
@petegoodliffe

pete@goodliffe.net



KEEP
CALM
AND
CARRY
ON

@petegoodliffe
pete@goodliffe.net
goodliffe.blogspot.com
www.goodliffe.net



i knew you'd want another

```
[UIView animateWithDuration:0.5
    animations:^(
        imageView.layer.opacity = 0.1;
    )
    completion:^(BOOL finished){
        [imageView removeFromSuperview];
    }
];
```

BUMPH DULL, but important



THIS DOCUMENT WAS CREATED BY PETE GOODLIFFE

IT IS COPYRIGHT // © 2011 PETE GOODLIFFE

>> ALL RIGHTS RESERVED

>> ALL THOUGHTS ARE OWNED

**>> PHOTOS AND IMAGES ARE MOSTLY
SOURCED FROM THE WEB**

THANK YOU FOR READING // I HOPE IT WAS USEFUL

Why C++ Sails When the Vasa Sank

Scott Meyers, Ph.D.
Software Development Consultant

smeyers@aristeia.com
<http://www.aristeia.com/>

Voice: 503/638-6028
Fax: 503/974-1887

Scott Meyers, Software Development Consultant
<http://www.aristeia.com/>

© 2011 Scott Meyers, all rights reserved.
Last Revised: 4/16/11

C++ in 1992

Endless stream of extension proposals.

Every extension proposal should be required to be accompanied by a kidney. People would submit only serious proposals, and nobody would submit more than two.

— Jim Waldo

C++ is already too large and complicated for our taste...

Remember the Vasa!

— Bjarne Stroustrup

Scott Meyers, Software Development Consultant
<http://www.aristeia.com/>

© 2011 Scott Meyers, all rights reserved.

Slide 2

The Vasa



- Commissioned 1625 by Swedish King Gustav.
- Much redesign during construction.
- Heavily armed and decorated.

Scott Meyers, Software Development Consultant
<http://www.aristeia.com/>

© 2011 Scott Meyers, all rights reserved.

Slide 3

The Vasa

Top-heavy and unstable, it sank on its maiden voyage.



Scott Meyers, Software Development Consultant
<http://www.aristeia.com/>

© 2011 Scott Meyers, all rights reserved.

Slide 4

C++: Heavily Armed

Classes (concrete & abstract)	Namespaces	Virtual member functions
Constructors	RTTI	Nested classes
Destructors	Locales	Nonvirtual functions
Default parameters	Strings	Static member functions
Operator overloading	The STL	Pure virtual functions
<code>new</code> & <code>delete</code>	Iostreams	Function overloading
Inheritance (public, private, protected, virtual, multiple)	Static data (in classes, functions, namespaces, files)	Templates (including total & partial specialization)
Nonvirtual member functions	Inline functions	Function overloading
Exceptions	References	New Handlers
Private & protected members	<code>bool</code>	Friends

Scott Meyers, Software Development Consultant
<http://www.aristeia.com/>

© 2011 Scott Meyers, all rights reserved.

Slide 5

C++: Ornately Decorated

What does this do?

```
f(x); // "invoke f on x"
```

- `f` may be a function (or function pointer or function reference).
- `f` may be an instance of a class overloading operator().
- `f` may be an object implicitly convertible to one of the above.
- `f` may be an overloaded function name.
 - ➔ At any of multiple scopes.
- `f` may be the name of one or more templates.
- `f` may be the name of one or more template specializations.
 - ➔ Total or partial.
- **`f` may be several of the above!**
 - ➔ E.g., overloaded function name + overloaded template name + name of template specialization(s).

Scott Meyers, Software Development Consultant
<http://www.aristeia.com/>

© 2011 Scott Meyers, all rights reserved.

Slide 6

C++ Must have Done Something Right

Programming Language	Position Feb 2011	Position Feb 2006	Position Feb 1996	Position Feb 1986
Java	1	1	-	-
C	2	2	1	1
C++	3	3	3	8
Python	4	8	22	-
PHP	5	5	-	-
C#	6	7	-	-
(Visual) Basic	7	4	2	5
Objective-C	8	46	-	-
JavaScript	9	10	-	-
Perl	10	6	7	-
Lisp	13	14	5	3
Ada	20	17	10	2

C++ Must have Done Something Right

Language use in open source (from Black Duck):

Rank	Language	%
1	C	42.15
2	C++	12.43
3	Javascript	9.27
4	Java	8.87
5	PHP	6.09
6	Shell	3.98
7	Ruby	3.18
8	Python	2.69
9	Perl	1.78
10	Assembler	1.56
11	SQL	1.53
12	C#	1.21
13	XML Schema	1.04
14	Ada	0.57
15	MetaFont	0.49

Compatibility with

- **Programmers:**
 - ➔ Important in 1982.
 - ➔ Important now.
 - ◆ C longevity astonishing.
 - **Source code:**
 - ➔ "As close as possible to C, but no closer."
 - **Object code:**
 - ➔ Direct access to universal assembly language.
 - **Build chains:**
 - ➔ Editors, compilers, linkers, debuggers, etc.
- C & C++ still often lumped together: "C/C++".**

Programming Language	Position Feb 2011	Position Feb 2006	Position Feb 1996	Position Feb 1988
Java	1	1	-	-
C++	2	2	1	1
C	3	3	3	8
Python	4	8	22	-
PHP	5	5	-	-
C#	6	7	-	-
(Visual) Basic	7	4	2	5
Objective-C	8	46	-	-
JavaScript	9	10	-	-
Perl	10	6	7	-
Lisp	13	14	5	3
Ada	20	17	10	2

Rank	Language	%
1	C	40.13
2	C++	13.43
3	JavaScript	9.27
4	Java	8.87
5	PHP	6.09
6	Shell	5.86
7	Ruby	3.18
8	Python	2.89
9	Perl	1.78
10	Assembly	1.56
11	SQL	1.53
12	C#	1.31
13	XML Schema	1.04
14	Ada	0.97
15	Matlab	0.49

Very General Features

Lead to surprising uses.

- **Destructors.**
 - ➔ RAII ⇒ general-purpose Undo.
 - ➔ Exception safety.
- **Templates.**
 - ➔ Generic programming (e.g., STL).
 - ➔ TMP.
 - ◆ Dimensional units analysis.
- **Overloading.**
 - ➔ Generic programming.
 - ➔ Smart pointers.
 - ➔ Function objects.



A language for library writers.

$$\frac{1}{X_0} = 4\alpha r_e^2 \frac{N_A}{A} \{ Z^2 [L_{rad} - f(Z)] + Z L'_{rad} \}$$

```
Energy<> finalEnergy(Element<> const & material, Density<> const dens,
                    Length<> const thick, Energy<> const initEnergy) {
    AtomicWeight<> const A = material->atomicWeight;
    AtomicNumber<> const Z = material->atomicNumber;
    Number<> const L_rad = log( 184.15 / root<3>( Z ) );
    Number<> const Lp_rad = log( 1194. / root<3>(Z*Z) );
    Length<> const X_0 = 4.0 * alpha * r_e * r_e * N_A / A *
        ( Z * Z * L_rad + Z * Lp_rad );
    return initEnergy / exp( thick / X_0 );
}
```

Paradigm Agnosticism



- **Procedural programming.**
 - ➔ Free functions:
 - ◆ Basis of STL.
 - ◆ Naturally models symmetric relationships.
 - ◆ Facilitates natural type extension.
- **OOP.**
 - ➔ Including MI.
- **Generic programming.**

I tried to implement STL in other languages and failed. C++ was the only language in which I could do it.

— Alexander Stepanov
- **Functional programming.**
 - ➔ Function objects.
 - ➔ TMP.

Paradigm Agnosticism

- **“Unsafe” programming.**
 - ➔ Pointers, casts, unchecked arrays, etc.
 - ➔ **“Trust the programmer.”**
- **“Safe” programming.**
 - ➔ Vectors with bounds checking, checked iterators, etc.



Commitment to Systems Programming

Suitable for challenging systems applications.

- **Program speed.**
 - ➔ Many systems can never be fast enough.
- **Program size.**
 - ➔ Static (including RTL).
 - ➔ Dynamic.
- **Data layout.**
 - ➔ Drivers.
 - ➔ Communications protocols.
 - ➔ Use with legacy code/systems.
- **Efficient communication with foreign entities.**
 - ➔ Hardware.
 - ➔ OSes.
 - ➔ Code in other languages.



C++ vs. The Vasa

So far, C++ has escaped the fate of the Vasa; it has not keeled over and disappeared – despite much wishful thinking and many dire predictions.

— Bjarne Stroustrup, 1996

Why?

- Compatibility with C.
- Very general features.
- Paradigm agnosticism.
- Commitment to systems programming.

Further Information

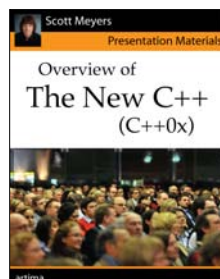
- “Vasa (ship),” *Wikipedia*.
- “The Vasa: A Disaster Story with Software [sic] Analogies,” Linda Rising, *The Software Practitioner*, January-February 2001.
- “How to write a C++ language extension proposal,” Bjarne Stroustrup, *The C++ Report*, May 1992.
- “C++: The Making of a Standard Journey's End,” Chuck Allison, *C/C++ Users Journal*, October 1996.
 - ➔ Primarily an interview with Bjarne Stroustrup.
- “TIOBE Programming Community Index,” *TIOBE Software*, <http://www.tiobe.com/index.php/content/paperinfo/tpci/>.
- “Open Source Project Data,” *Black Duck Open Source Resource Center*, <http://www.blackducksoftware.com/oss/projects>.
- “C/C++'s Enduring Popularity,” *Coverity Software Integrity Blog*, February 22, 2010.

Licensing Information

Scott Meyers licenses materials for this and other training courses for commercial or personal use. Details:

- **Commercial use:** <http://aristeia.com/Licensing/licensing.html>
- **Personal use:** <http://aristeia.com/Licensing/personalUse.html>

Courses currently available for personal use include:



About Scott Meyers



Scott is a trainer and consultant on the design and implementation of software systems, typically in C++. His web site,

<http://www.aristeia.com/>

provides information on:

- Training and consulting services
- Books, articles, other publications
- Upcoming presentations
- Professional activities blog