Lightning Talks

accu 2011

Kevlin Henney – Not So SOLID Crew

Mike Long – Unit Testing Legacy C

Didier Verna – Letting Go Of Control (part 2)

Paul Black – Thimama Merodia: A New Authentication Mechanism

Tom Gilb - Simplicity

Matty Williams – I'm Not A Doctor, Trust Me

Didier Verna - (untitled)

Charles Bailey – Massaging Hunks

Seb Rose – Referential Integrity

Richard Harris – Comparing Floats

Jim Hague – Divine Guidance

Bjorn Eriksson – Some Things We Learned...

Phil Nash – CATCH

Lightning Talks

Kevlin Henney – Not So SOLID Crew

Mike Long – Unit Testing Legacy C

Didier Verna – Letting Go Of Control (part 2)

Paul Black – Thimama Merodia: A New Authentication Mechanism

Tom Gilb - Simplicity

Matty Williams – I'm Not A Doctor, Trust Me

Didier Verna - (untitled)

Charles Bailey – Massaging Hunks

Seb Rose – Referential Integrity

Richard Harris – Comparing Floats

Jim Hague – Divine Guidance

Bjorn Eriksson – Some Things We Learned...

Phil Nash – CATCH

Not So SOLID Crew

Kevlin Henney
kevlin@curbralan.com
@KevlinHenney

How solid are the SOLID principles?

principle

- *a fundamental truth or proposition that serves as the foundation for a system of belief or behaviour or for a chain of reasoning.*
- *morally correct behaviour and attitudes.*
- *a general scientific theorem or law that has numerous special applications across a wide field.*
- *a natural law forming the basis for the construction or working of a machine.*

Oxford Dictionary of English

Single Responsibility

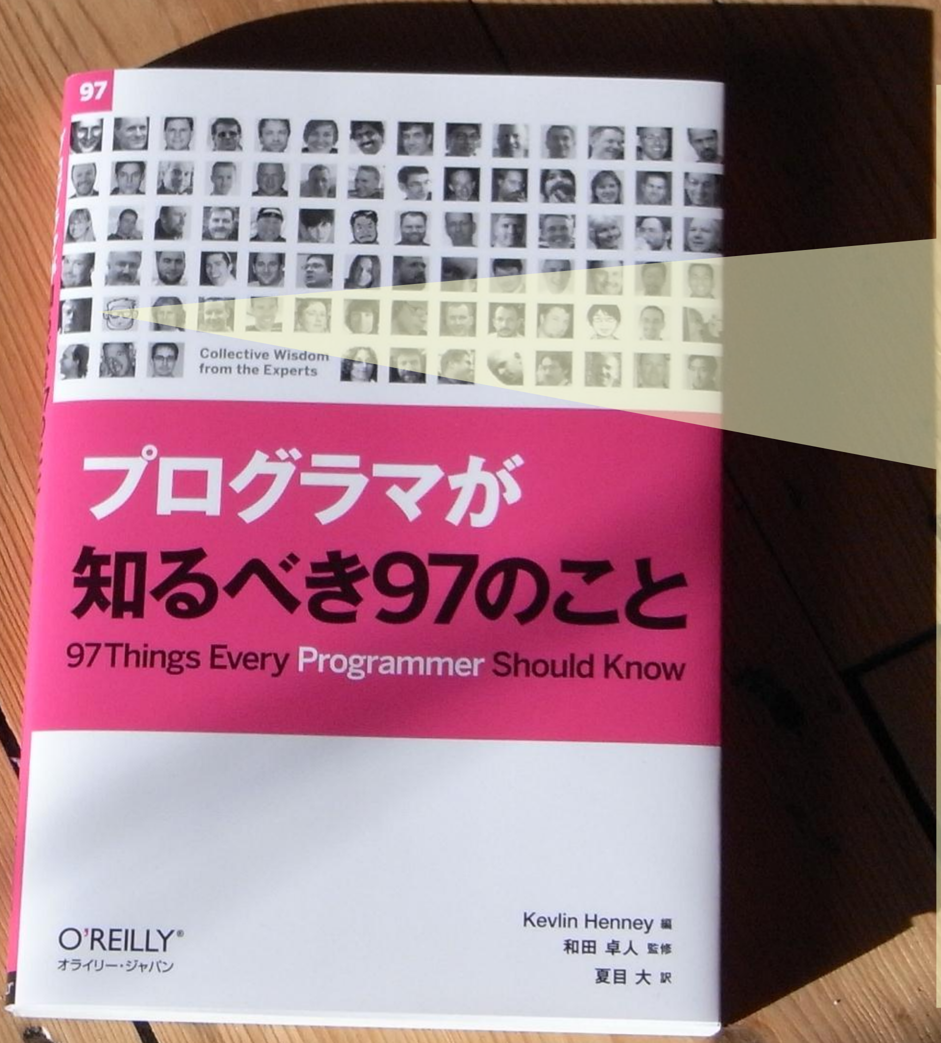
Open-Closed

Liskov Substitution

Interface Segregation

Dependency Inversion

97
Collective Wisdom from the Experts
プログラマが知るべき97のこと
97 Things Every Programmer Should Know
O'REILLY
オライリー・ジャパン
Kevlin Henney 編
和田 卓人 監修
夏目 大 訳


97
Collective Wisdom from the Experts
プログラマが知るべき97のこと
97 Things Every Programmer Should Know
O'REILLY®
オライリー・ジャパン
Kevlin Henney 編
和田 卓人 監修
夏目 大 訳

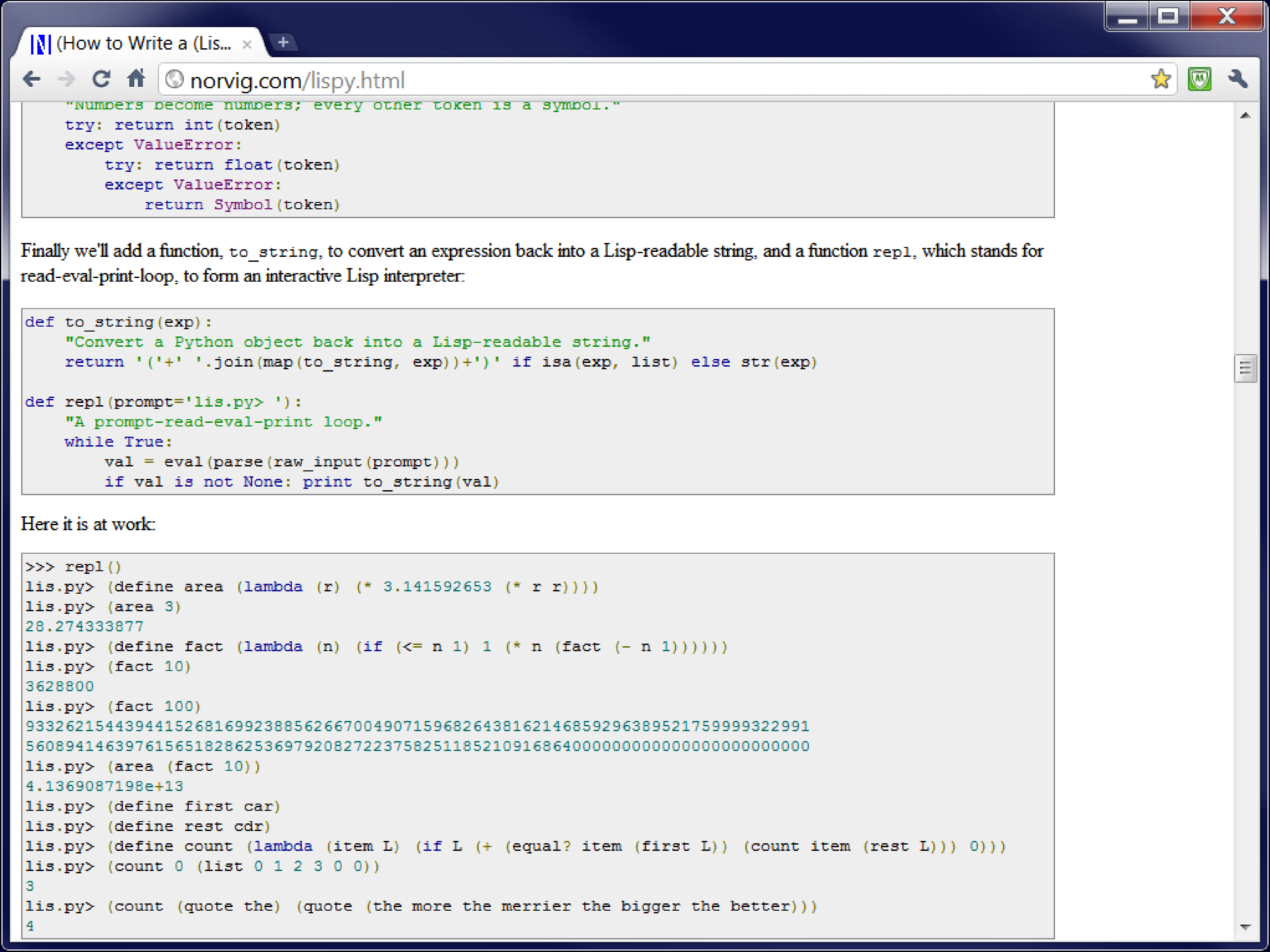
One of the most foundational principles of good design is:

> Gather together those things that change for the same reason, and separate those things that change for different reasons.

This principle is often known as the *single responsibility principle*, or SRP. In short, it says that a subsystem, module, class, or even a function, should not have more than one reason to change.

Every class should embody only about 3–5 distinct responsibilities.

Grady Booch, *Object Solutions*



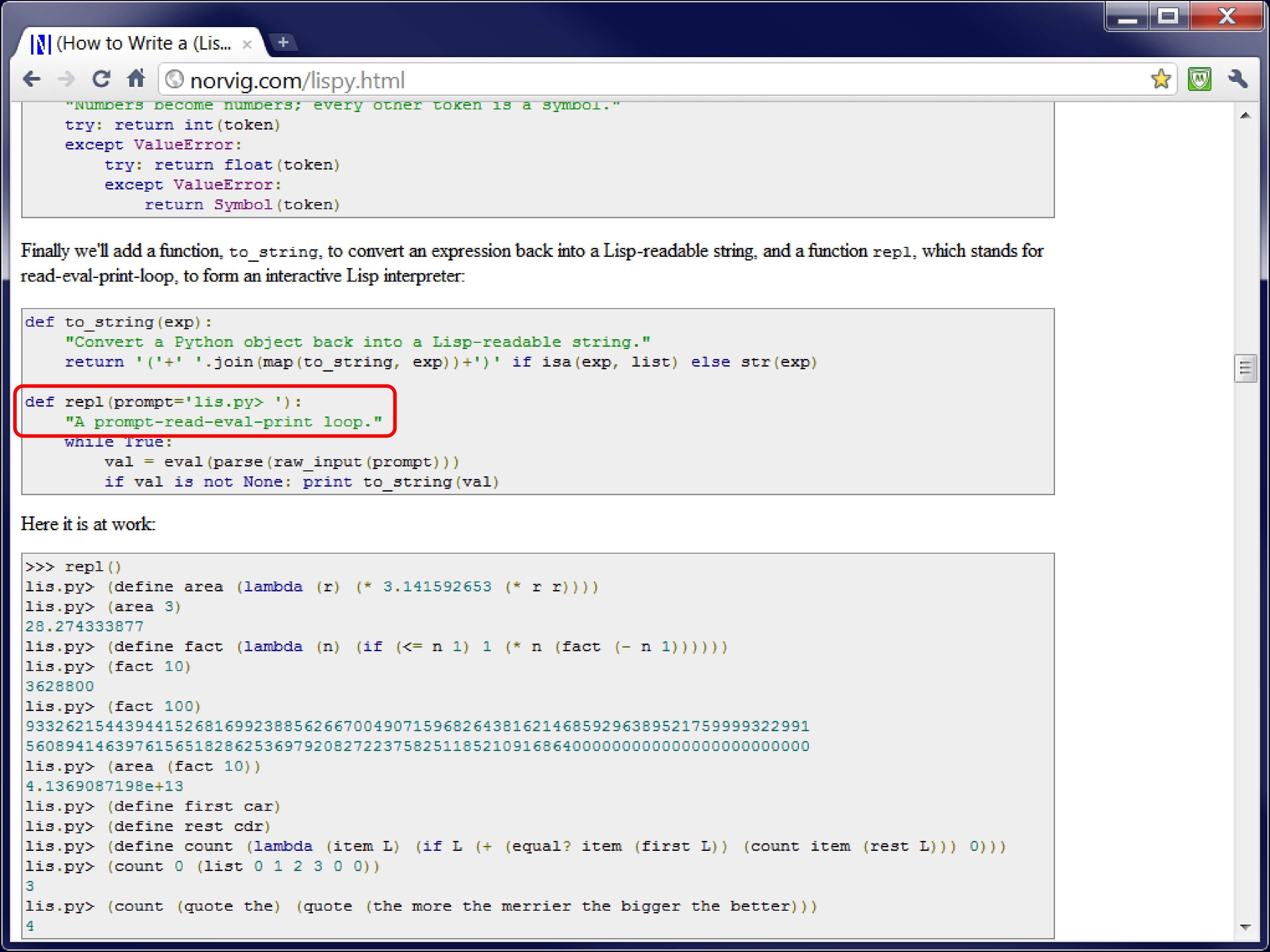
```
    "Numbers become numbers; every other token is a symbol."
    try: return int(token)
    except ValueError:
        try: return float(token)
        except ValueError:
            return Symbol(token)
```

Finally we'll add a function, `to_string`, to convert an expression back into a Lisp-readable string, and a function `repl`, which stands for read-eval-print-loop, to form an interactive Lisp interpreter:

```
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)
```

Here it is at work:

```
>>> repl()
lis.py> (define area (lambda (r) (* 3.141592653 (* r r))))
lis.py> (area 3)
28.274333877
lis.py> (define fact (lambda (n) (if (<= n 1) 1 (* n (fact (- n 1))))))
lis.py> (fact 10)
3628800
lis.py> (fact 100)
9332621544394415268169923885626670049071596826438162146859296389521759999322991
5608941463976156518286253697920827223758251185210916864000000000000000000000000
lis.py> (area (fact 10))
4.1369087198e+13
lis.py> (define first car)
lis.py> (define rest cdr)
lis.py> (define count (lambda (item L) (if L (+ (equal? item (first L)) (count item (rest L))) 0)))
lis.py> (count 0 (list 0 1 2 3 0 0))
3
lis.py> (count (quote the) (quote (the more the merrier the bigger the better)))
4
```



```
    "Numbers become numbers; every other token is a symbol."
    try: return int(token)
    except ValueError:
        try: return float(token)
        except ValueError:
            return Symbol(token)
```

Finally we'll add a function, `to_string`, to convert an expression back into a Lisp-readable string, and a function `repl`, which stands for read-eval-print-loop, to form an interactive Lisp interpreter:

```
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)
```

Here it is at work:

```
>>> repl()
lis.py> (define area (lambda (r) (* 3.141592653 (* r r))))
lis.py> (area 3)
28.274333877
lis.py> (define fact (lambda (n) (if (<= n 1) 1 (* n (fact (- n 1))))))
lis.py> (fact 10)
3628800
lis.py> (fact 100)
9332621544394415268169923885626670049071596826438162146859296389521759999322991
5608941463976156518286253697920827223758251185210916864000000000000000000000000
lis.py> (area (fact 10))
4.1369087198e+13
lis.py> (define first car)
lis.py> (define rest cdr)
lis.py> (define count (lambda (item L) (if L (+ (equal? item (first L)) (count item (rest L))) 0)))
lis.py> (count 0 (list 0 1 2 3 0 0))
3
lis.py> (count (quote the) (quote (the more the merrier the bigger the better)))
4
```

~~Single Responsibility~~

Open-Closed

Liskov Substitution

Interface Segregation

Dependency Inversion

The principle stated that a good module structure should be both open and closed:

- Closed, because clients need the module's services to proceed with their own development, and once they have settled on a version of the module should not be affected by the introduction of new services they do not need.
- Open, because there is no guarantee that we will include right from the start every service potentially useful to some client.

Bertrand Meyer
Object-Oriented Software Construction

~~Single Responsibility~~

~~Open-Closed~~

Liskov Substitution

Interface Segregation

Dependency Inversion

A type hierarchy is composed of subtypes and supertypes. The intuitive idea of a subtype is one whose objects provide all the behavior of objects of another type (the supertype) plus something extra. What is wanted here is something like the following substitution property: If for each object o1 of type S there is an object o2 of type T such that for all programs P defined in terms of T, the behavior of P is unchanged when o1 is substituted for o2, then S is a subtype of T.

Barbara Liskov
"Data Abstraction and Hierarchy"

A type hierarchy is composed of subtypes and supertypes. The intuitive idea of a subtype is one whose objects provide all the behavior of objects of another type (the supertype) plus something extra. **What is wanted here is something like the following substitution property**: If for each object o1 of type S there is an object o2 of type T such that for all programs P defined in terms of T, the behavior of P is unchanged when o1 is substituted for o2, then S is a subtype of T.

Barbara Liskov
"Data Abstraction and Hierarchy"

A type hierarchy is composed of subtypes and supertypes. The intuitive idea of a subtype is one whose objects provide all the behavior of objects of another type (the supertype) plus something extra. What is wanted here is something like the following substitution property: If for each object o1 of type S there is an object o2 of type T such that for all programs P defined in terms of T, the behavior of P is unchanged when o1 is substituted for o2, then S is a subtype of T.

Barbara Liskov
"Data Abstraction and Hierarchy"

A type hierarchy is composed of subtypes and supertypes. The intuitive idea of a subtype is one whose objects provide all the behavior of objects of another type (the supertype) plus something extra. What is wanted here is something like the following substitution property: If for each object o1 of type S there is an object o2 of type T such that for all programs P defined in terms of T, the behavior of P is **unchanged** when o1 is substituted for o2, then S is a subtype of T.

Barbara Liskov
"Data Abstraction and Hierarchy"

~~Single Responsibility~~

~~Open-Closed~~

~~Liskov Substitution~~

Interface Segregation

Dependency Inversion

id

- Psychoanalysis *the part of the mind in which innate instinctive impulses and primary processes are manifest.*

Oxford Dictionary of English

id

- Psychoanalysis *the part of the mind in which innate instinctive impulses and primary processes are manifest.*

Oxford Dictionary of English

psychoanalysis

- *Freudian masturbation.*
- *Set of very strange ideas about female sexuality.*
- *Some pretty strange ideas about male sexuality.*
- *The reason your childhood has ruined the rest of your life.*

Urban Dictionary

Lightning Talks

accu 2011

Kevlin Henney – Not So SOLID Crew

Mike Long – Unit Testing Legacy C

Didier Verna – Letting Go Of Control (part 2)

Paul Black – Thimama Merodia: A New Authentication Mechanism

Tom Gilb - Simplicity

Matty Williams – I'm Not A Doctor, Trust Me

Didier Verna - (untitled)

Charles Bailey – Massaging Hunks

Seb Rose – Referential Integrity

Richard Harris – Comparing Floats

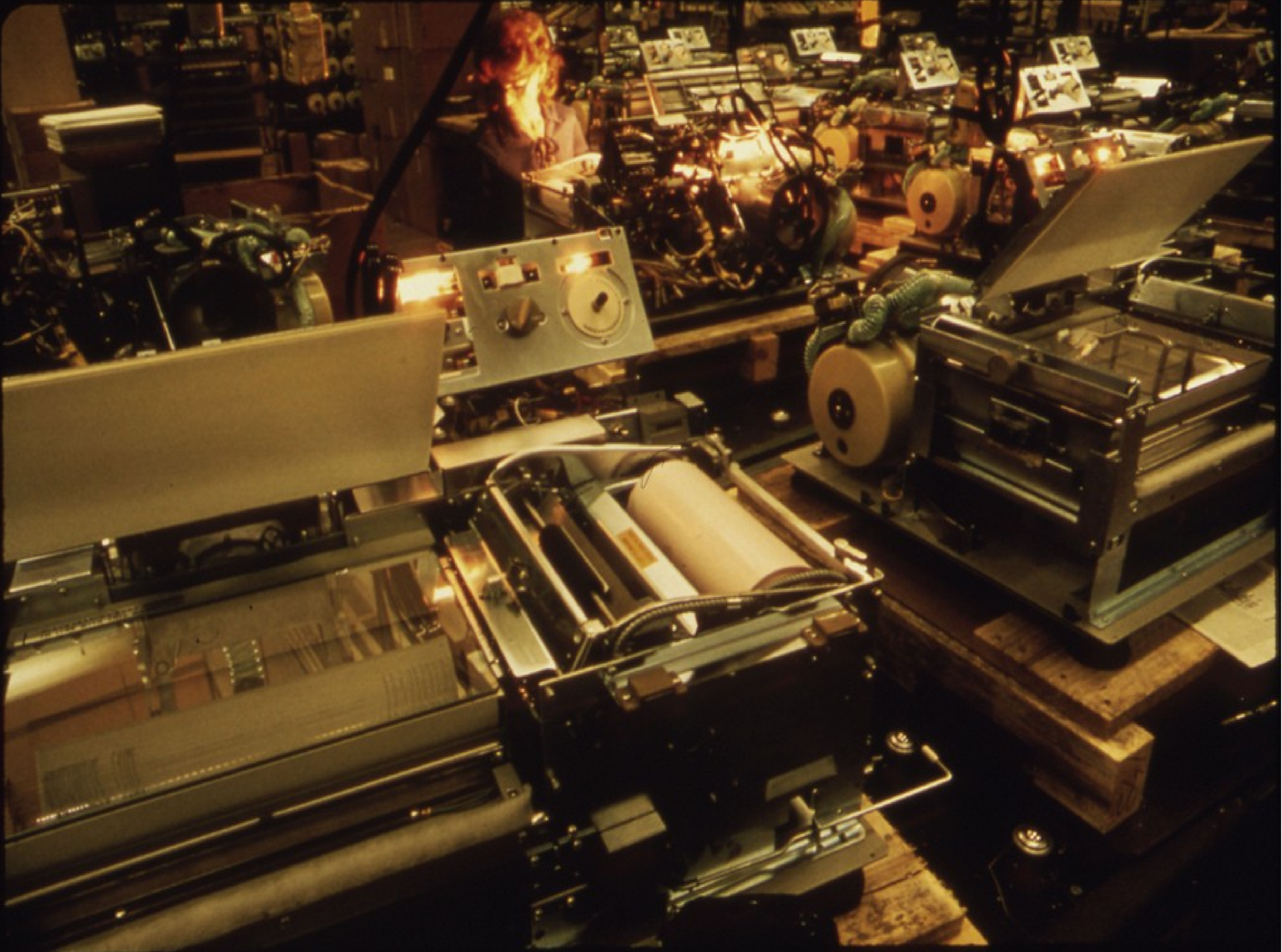
Jim Hague – Divine Guidance

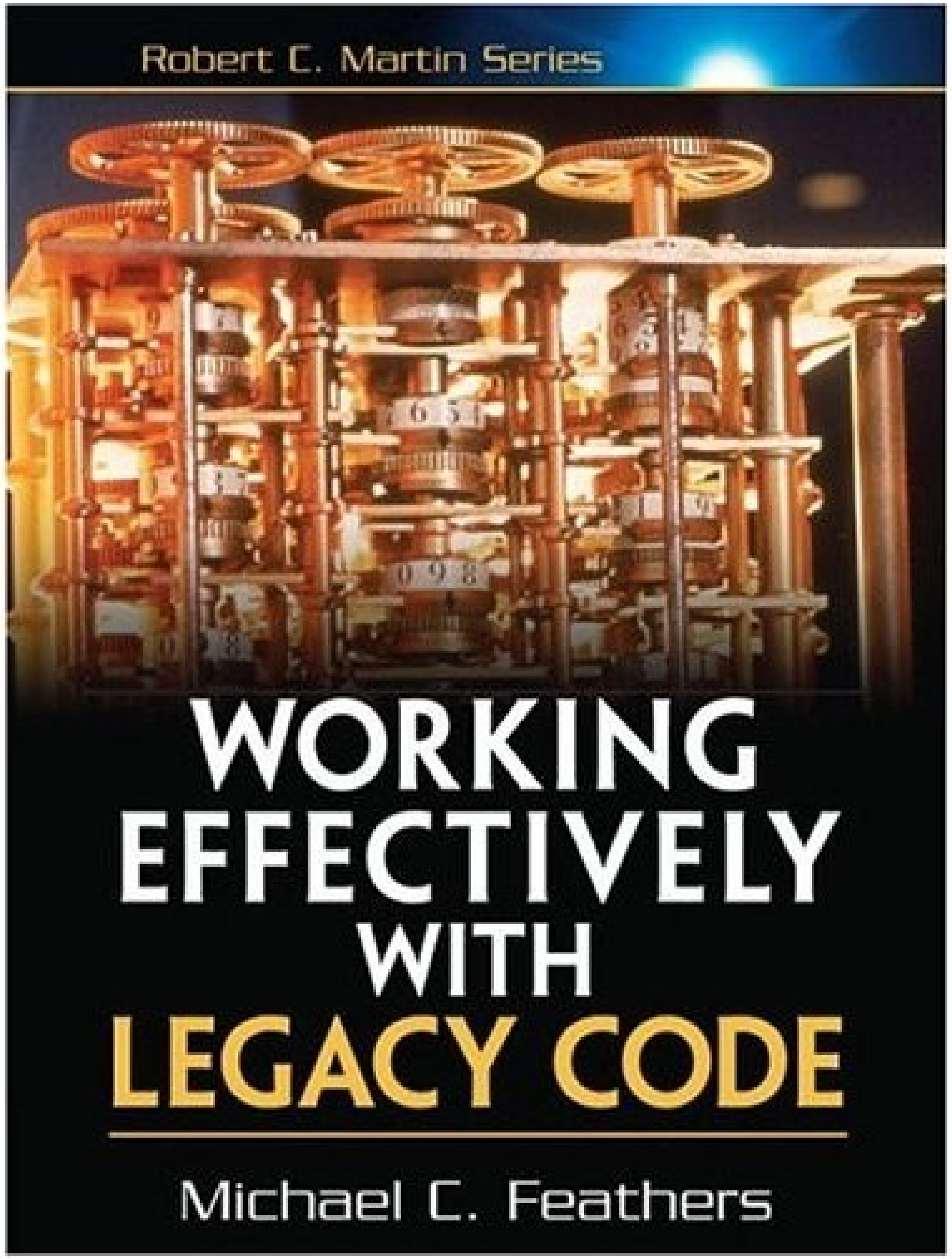
Bjorn Eriksson – Some Things We Learned...

Phil Nash – CATCH

Unit Testing Legacy C

Mike Long ACCU 2011



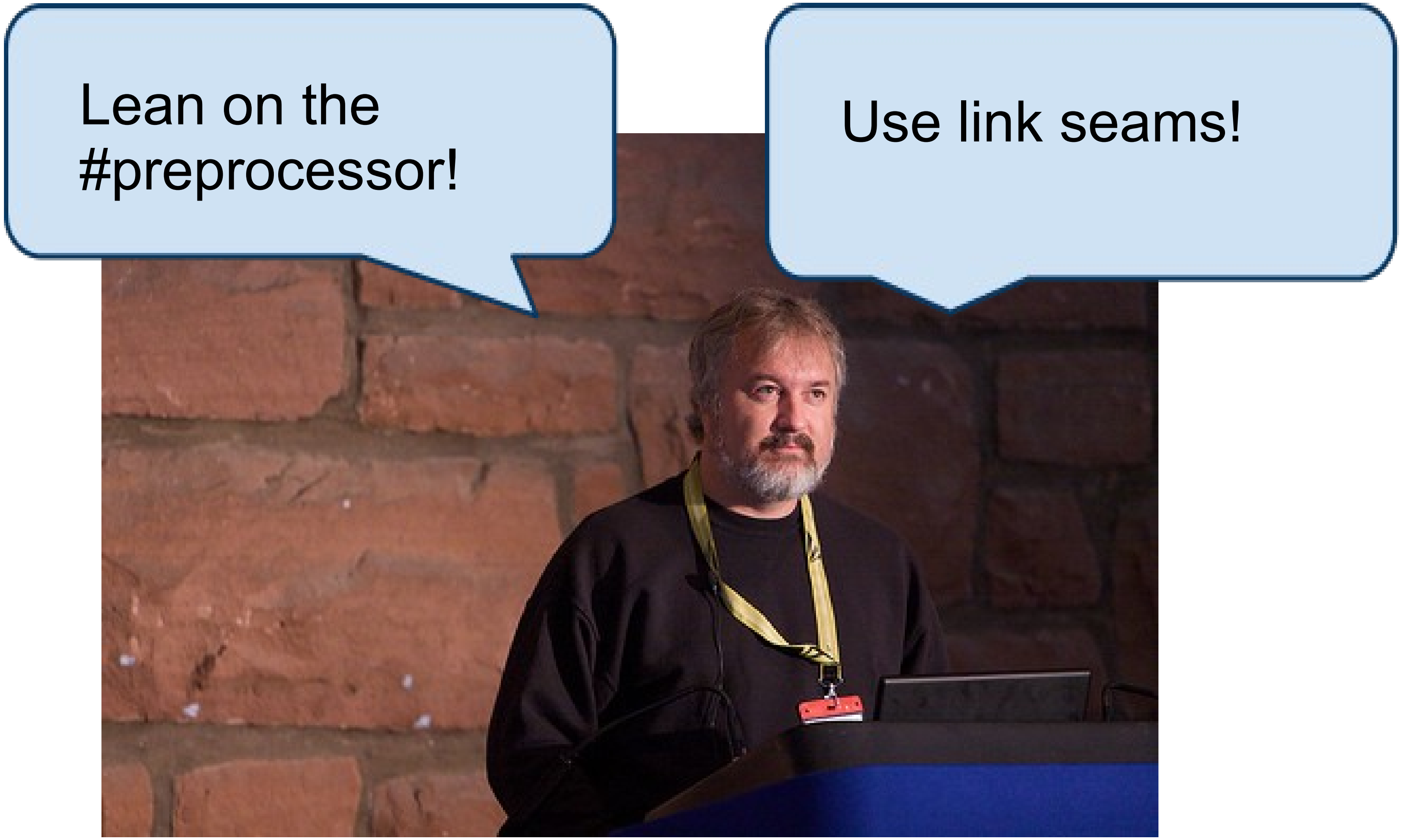


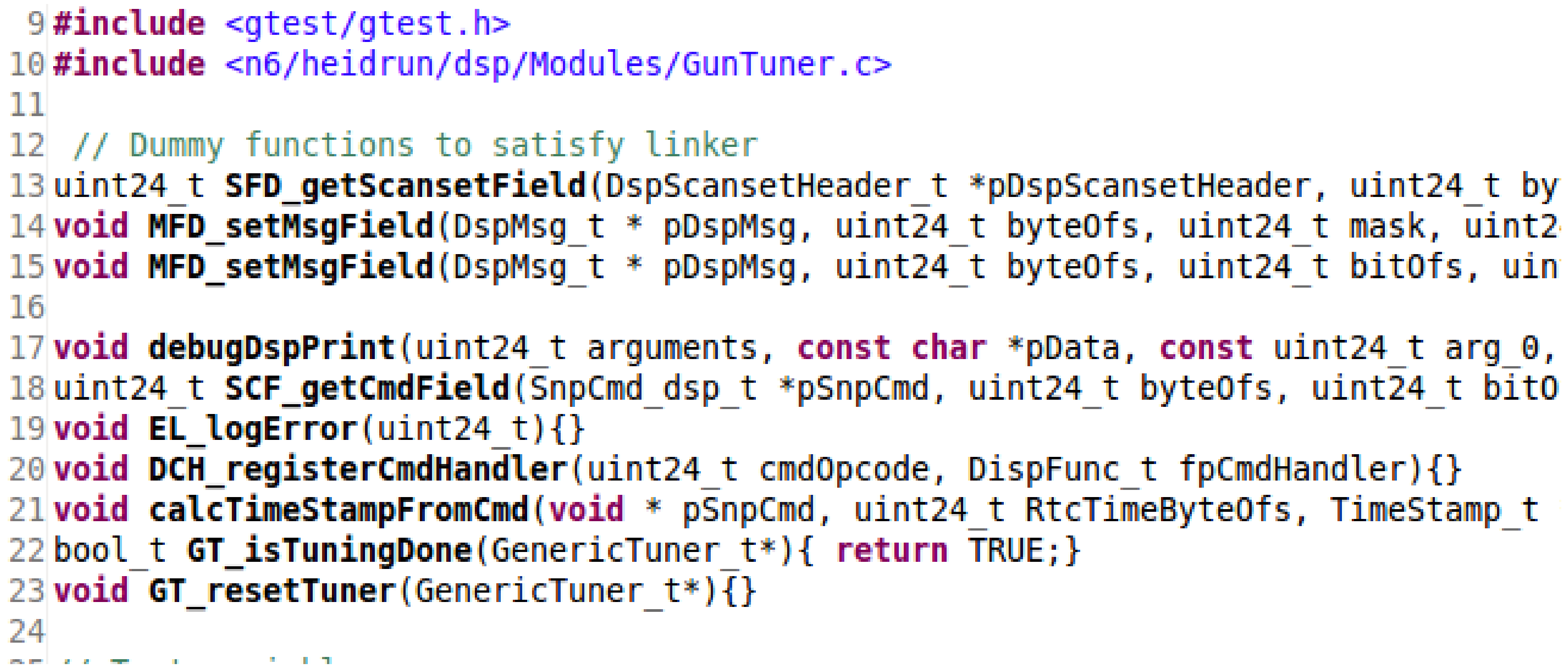
Robert C. Martin Series

WORKING EFFECTIVELY WITH LEGACY CODE

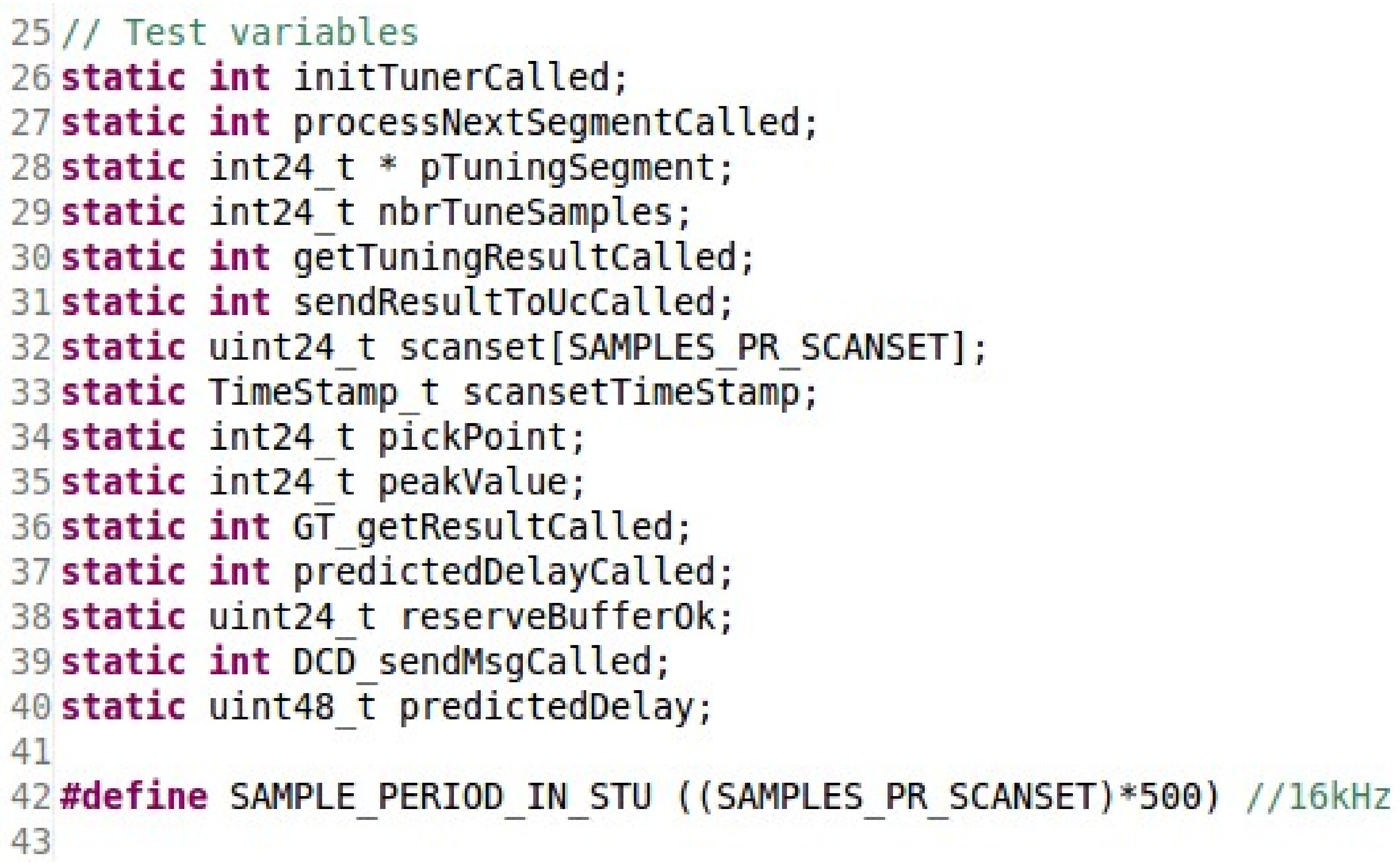
Michael C. Feathers

Use link seams!

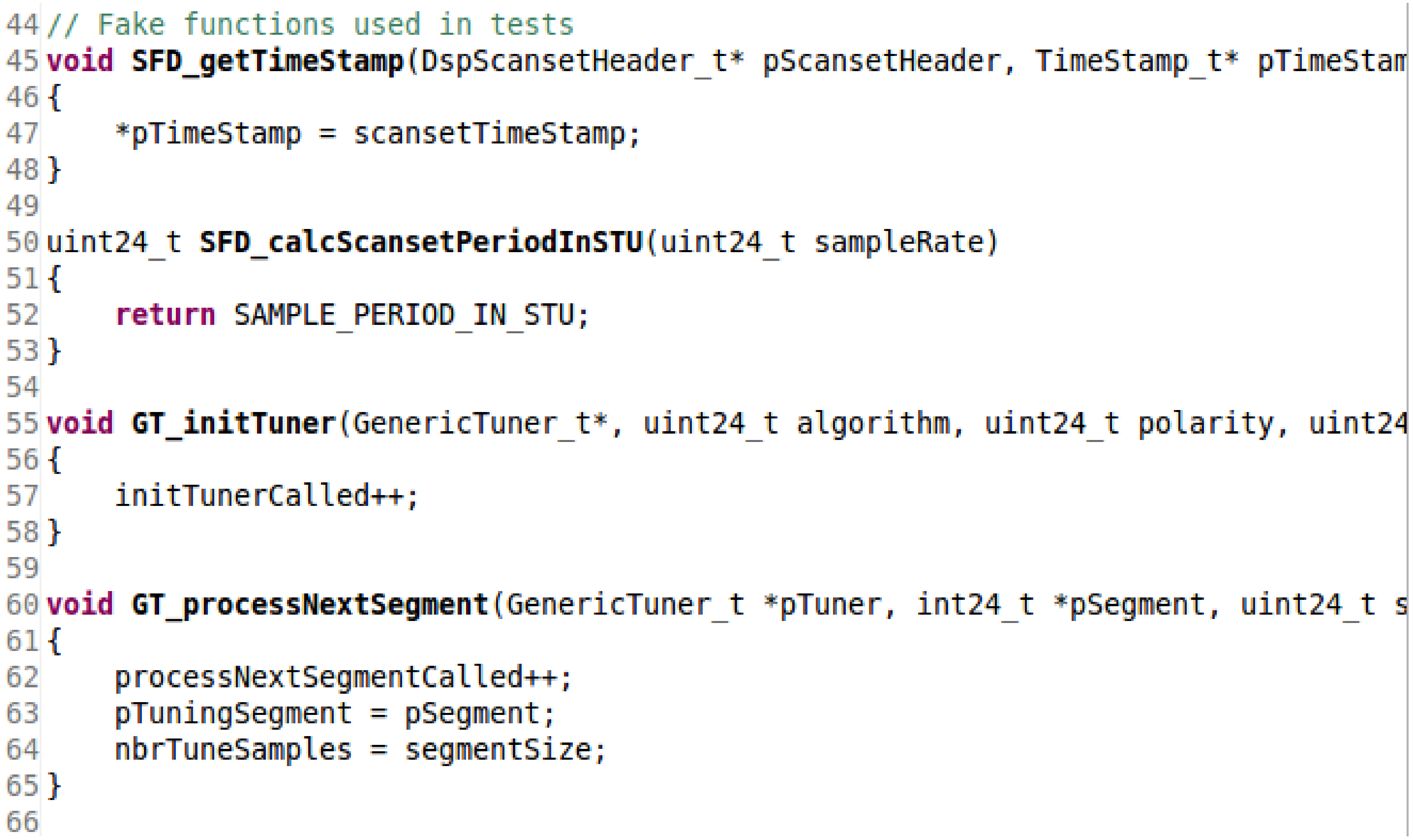

Lean on the #preprocessor!
Use link seams!



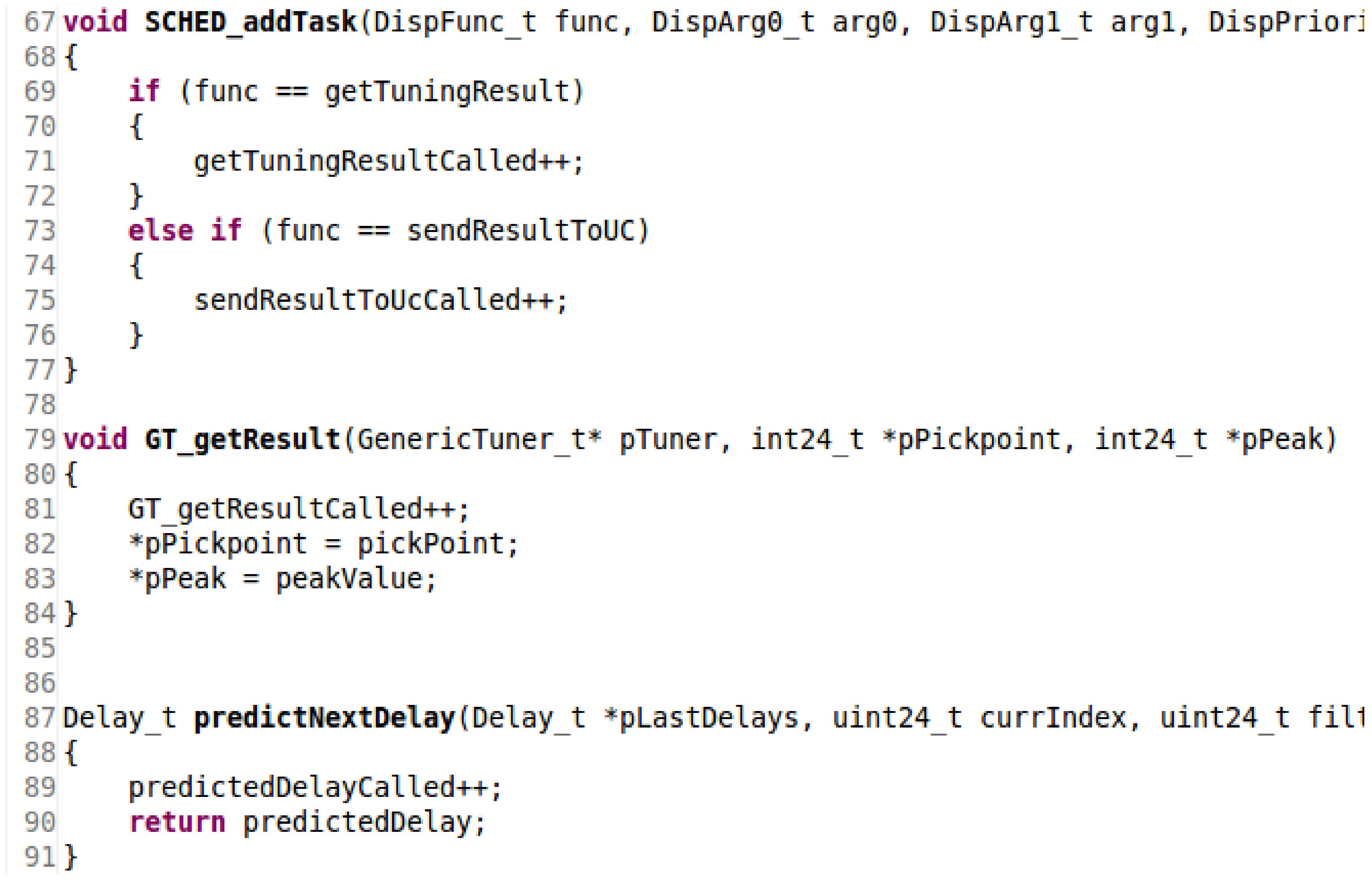
```
#include <gtest/gtest.h>
#include <n6/heidrun/dsp/Modules/GunTuner.c>

 // Dummy functions to satisfy linker
uint24_t SFD_getScansetField(DspScansetHeader_t *pDspScansetHeader, uint24_t by
void MFD_setMsgField(DspMsg_t * pDspMsg, uint24_t byteOfs, uint24_t mask, uint2
void MFD_setMsgField(DspMsg_t * pDspMsg, uint24_t byteOfs, uint24_t bitOfs, uin

void debugDspPrint(uint24_t arguments, const char *pData, const uint24_t arg_0,
uint24_t SCF_getCmdField(SnpCmd_dsp_t *pSnpCmd, uint24_t byteOfs, uint24_t bitO
void EL_logError(uint24_t){}
void DCH_registerCmdHandler(uint24_t cmdOpcode, DispFunc_t fpCmdHandler){}
void calcTimeStampFromCmd(void * pSnpCmd, uint24_t RtcTimeByteOfs, TimeStamp_t
bool_t GT_isTuningDone(GenericTuner_t*){ return TRUE;}
void GT_resetTuner(GenericTuner_t*){}

[illegible]
```



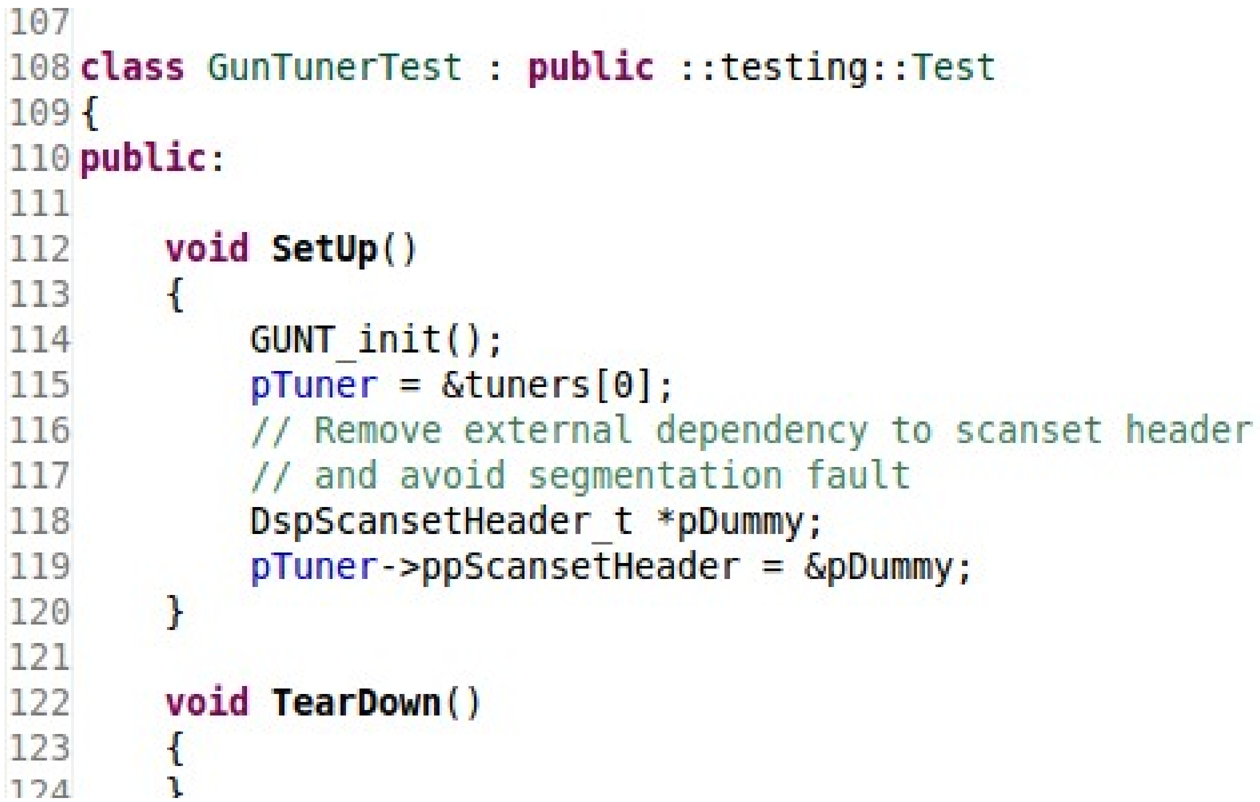
```c
// Test variables
static int initTunerCalled;
static int processNextSegmentCalled;
static int24_t * pTuningSegment;
static int24_t nbrTuneSamples;
static int getTuningResultCalled;
static int sendResultToUcCalled;
static uint24_t scanset[SAMPLES_PR_SCANSET];
static TimeStamp_t scansetTimeStamp;
static int24_t pickPoint;
static int24_t peakValue;
static int GT_getResultCalled;
static int predictedDelayCalled;
static uint24_t reserveBufferOk;
static int DCD_sendMsgCalled;
static uint48_t predictedDelay;

#define SAMPLE_PERIOD_IN_STU ((SAMPLES_PR_SCANSET)*500) //16kHz

```



```c
// Fake functions used in tests
void SFD_getTimeStamp(DspScansetHeader_t* pScansetHeader, TimeStamp_t* pTimeStam
{
    *pTimeStamp = scansetTimeStamp;
}

uint24_t SFD_calcScansetPeriodInSTU(uint24_t sampleRate)
{
    return SAMPLE_PERIOD_IN_STU;
}

void GT_initTuner(GenericTuner_t*, uint24_t algorithm, uint24_t polarity, uint24
{
    initTunerCalled++;
}

void GT_processNextSegment(GenericTuner_t *pTuner, int24_t *pSegment, uint24_t s
{
    processNextSegmentCalled++;
    pTuningSegment = pSegment;
    nbrTuneSamples = segmentSize;
}

```



```
void SCHED_addTask(DispFunc_t func, DispArg0_t arg0, DispArg1_t arg1, DispPriori
{
    if (func == getTuningResult)
    {
        getTuningResultCalled++;
    }
    else if (func == sendResultToUC)
    {
        sendResultToUcCalled++;
    }
}

void GT_getResult(GenericTuner_t* pTuner, int24_t *pPickpoint, int24_t *pPeak)
{
    GT_getResultCalled++;
    *pPickpoint = pickPoint;
    *pPeak = peakValue;
}


Delay_t predictNextDelay(Delay_t *pLastDelays, uint24_t currIndex, uint24_t filt
{
    predictedDelayCalled++;
    return predictedDelay;
}
```



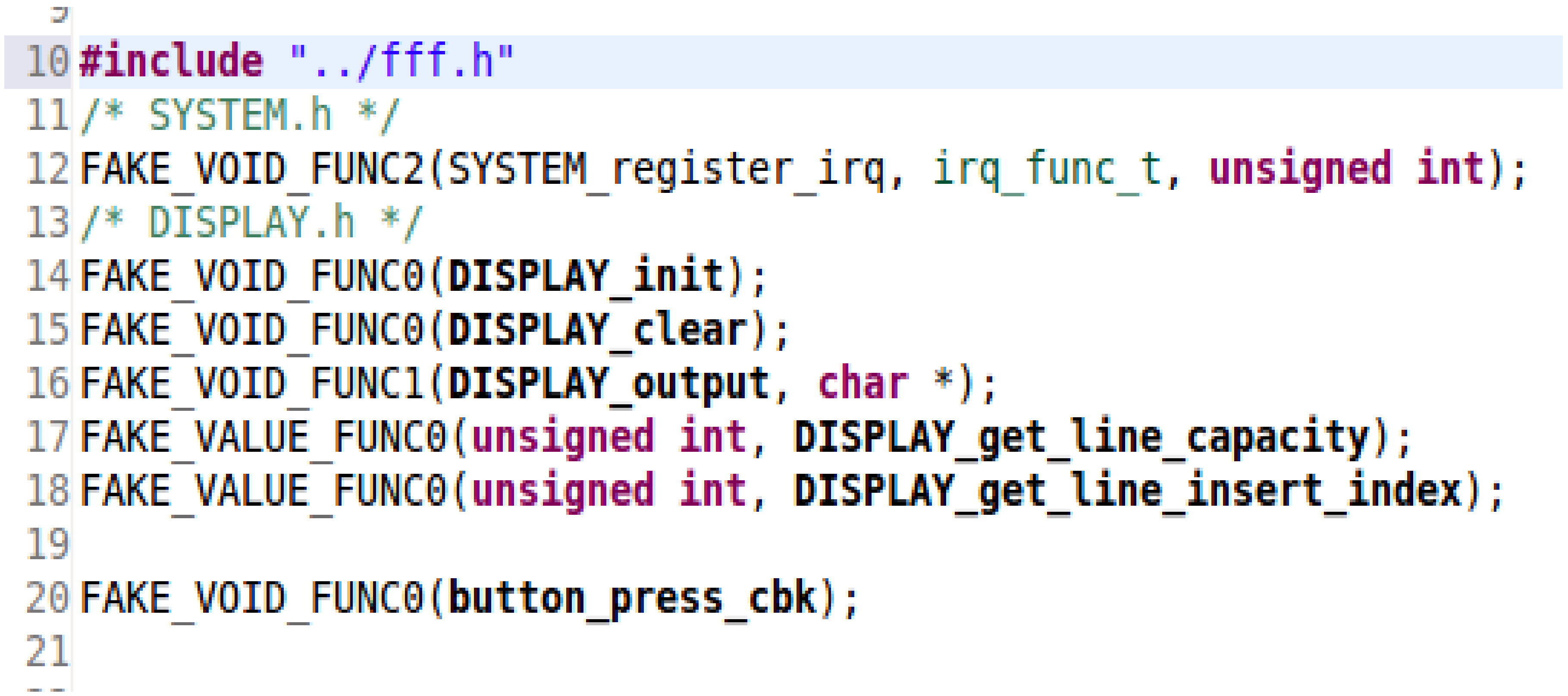
```

class GunTunerTest : public ::testing::Test
{
public:

    void SetUp()
    {
        GUNT_init();
        pTuner = &tuners[0];
        // Remove external dependency to scanset header
        // and avoid segmentation fault
        DspScansetHeader_t *pDummy;
        pTuner->ppScansetHeader = &pDummy;
    }

    void TearDown()
    {
    }
```

Is there a better way?

Fake Function Framework

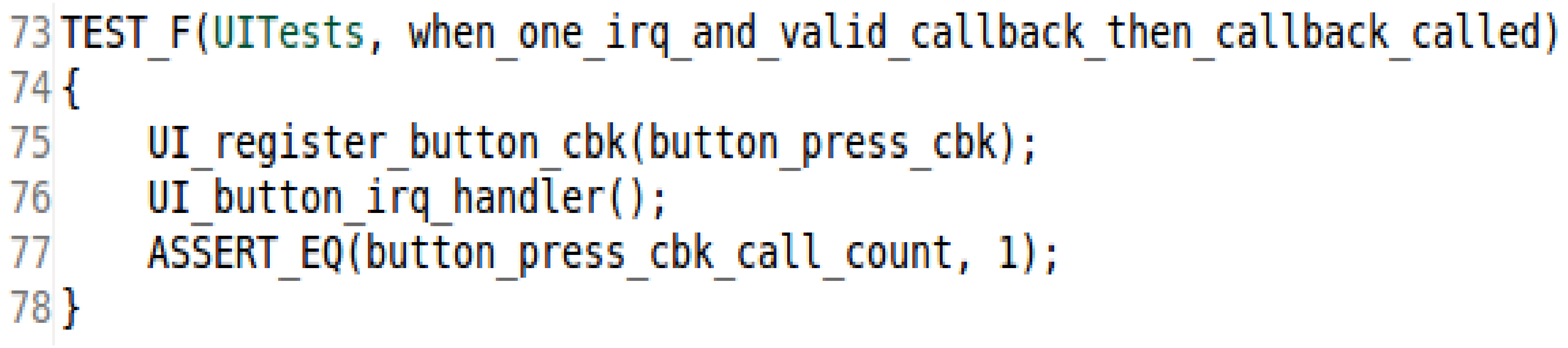
https://github.com/meekrosoft/fff

For Fake's Sake



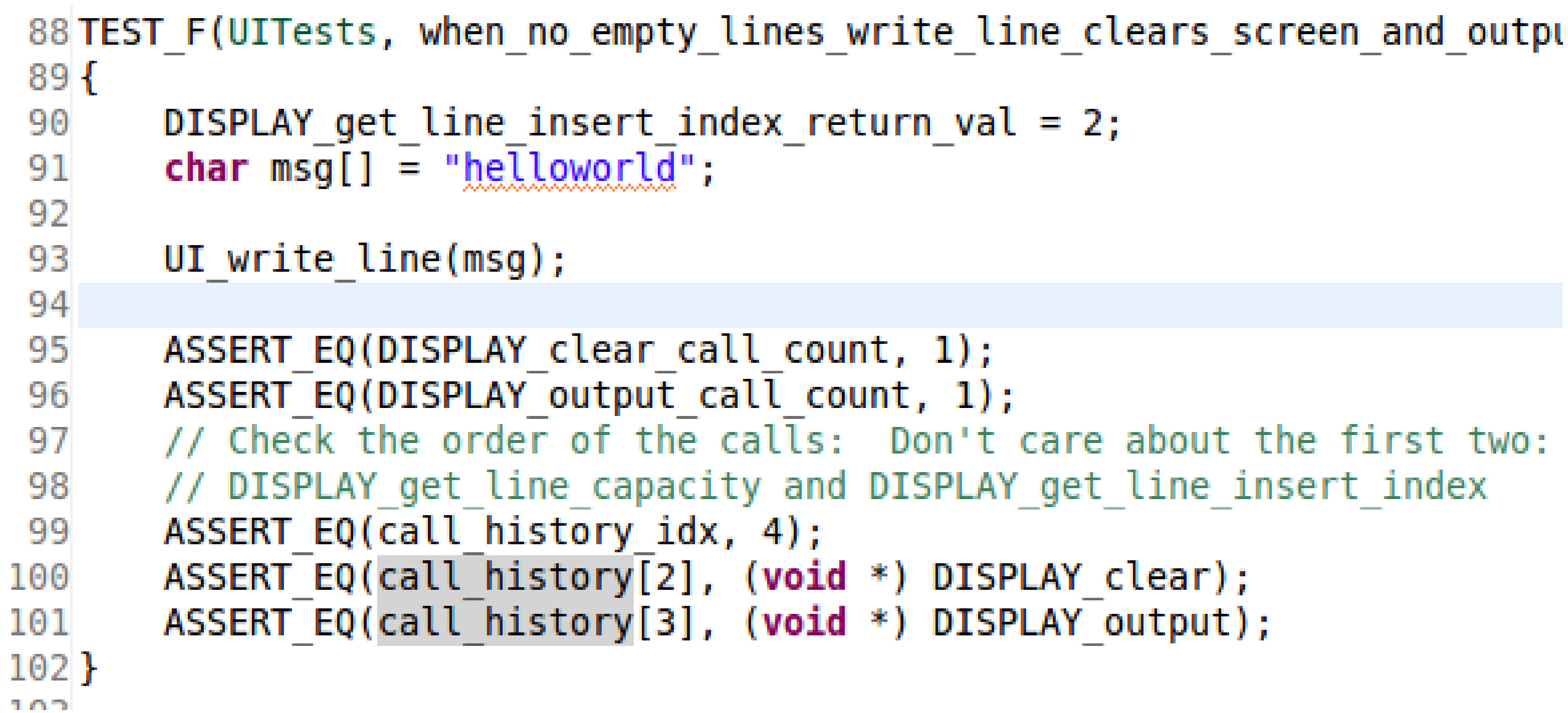
```c
#include "../fff.h"
/* SYSTEM.h */
FAKE_VOID_FUNC2(SYSTEM_register_irq, irq_func_t, unsigned int);
/* DISPLAY.h */
FAKE_VOID_FUNC0(DISPLAY_init);
FAKE_VOID_FUNC0(DISPLAY_clear);
FAKE_VOID_FUNC1(DISPLAY_output, char *);
FAKE_VALUE_FUNC0(unsigned int, DISPLAY_get_line_capacity);
FAKE_VALUE_FUNC0(unsigned int, DISPLAY_get_line_insert_index);

FAKE_VOID_FUNC0(button_press_cbk);

```

Function Call Counts!



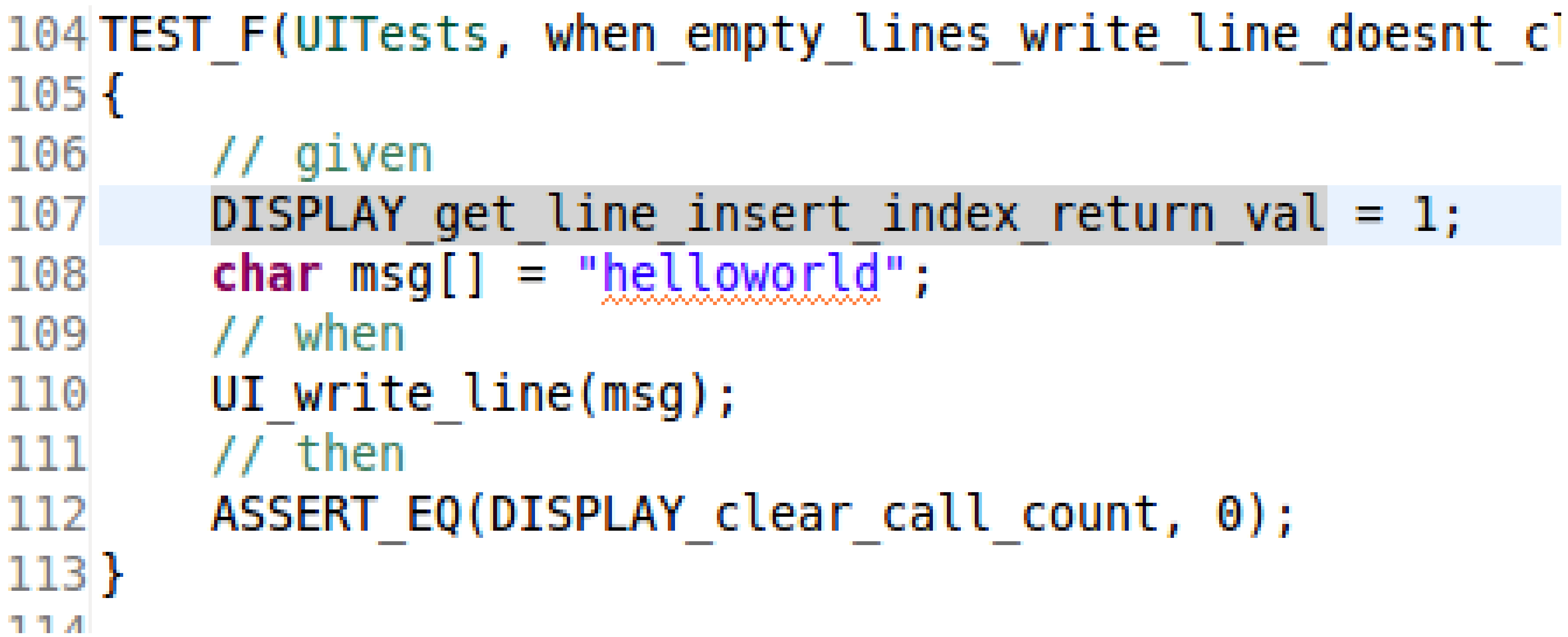
```
TEST_F(UITests, when_one_irq_and_valid_callback_then_callback_called)
{
    UI_register_button_cbk(button_press_cbk);
    UI_button_irq_handler();
    ASSERT_EQ(button_press_cbk_call_count, 1);
}
```

Function Call Sequence History!



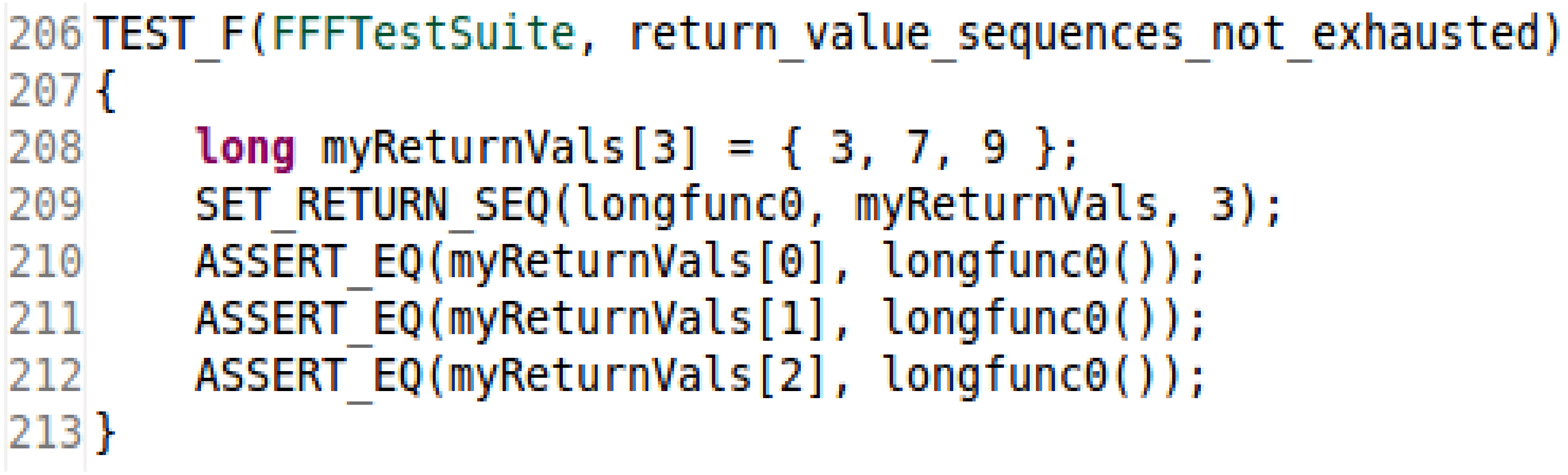
```
TEST_F(UITests, when_no_empty_lines_write_line_clears_screen_and_outpu
{
    DISPLAY_get_line_insert_index_return_val = 2;
    char msg[] = "helloworld";

    UI_write_line(msg);

    ASSERT_EQ(DISPLAY_clear_call_count, 1);
    ASSERT_EQ(DISPLAY_output_call_count, 1);
    // Check the order of the calls:  Don't care about the first two:
    // DISPLAY_get_line_capacity and DISPLAY_get_line_insert_index
    ASSERT_EQ(call_history_idx, 4);
    ASSERT_EQ(call_history[2], (void *) DISPLAY_clear);
    ASSERT_EQ(call_history[3], (void *) DISPLAY_output);
}
```

Specify Return Values!

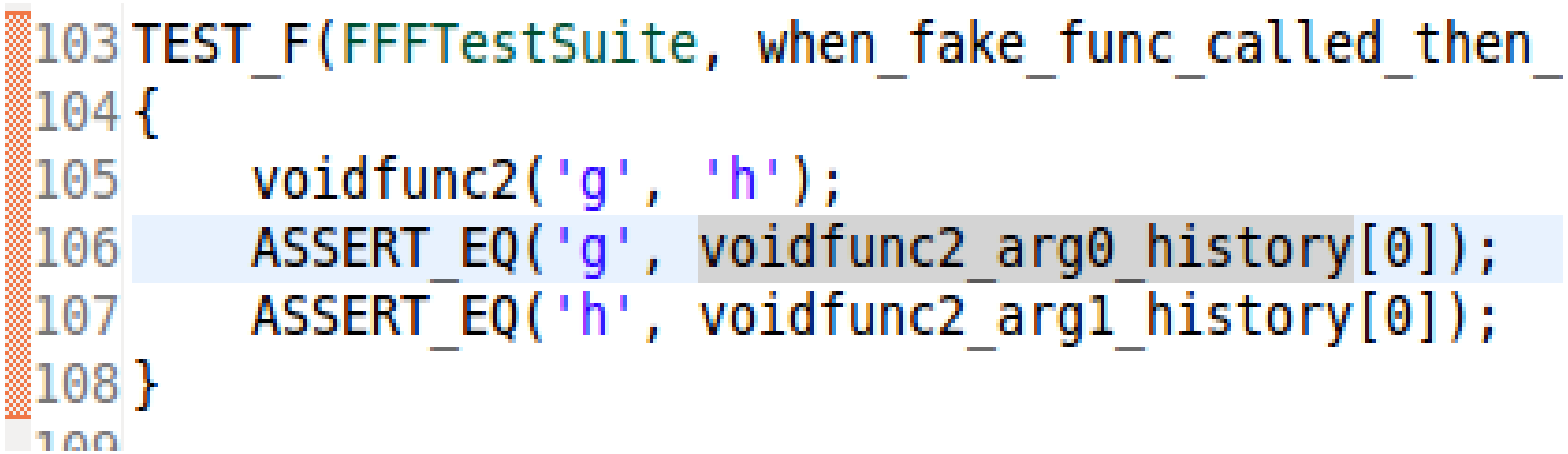


```
TEST_F(UITests, when_empty_lines_write_line_doesnt_c
{
    // given
    DISPLAY_get_line_insert_index_return_val = 1;
    char msg[] = "helloworld";
    // when
    UI_write_line(msg);
    // then
    ASSERT_EQ(DISPLAY_clear_call_count, 0);
}
```

Return Value Sequences!

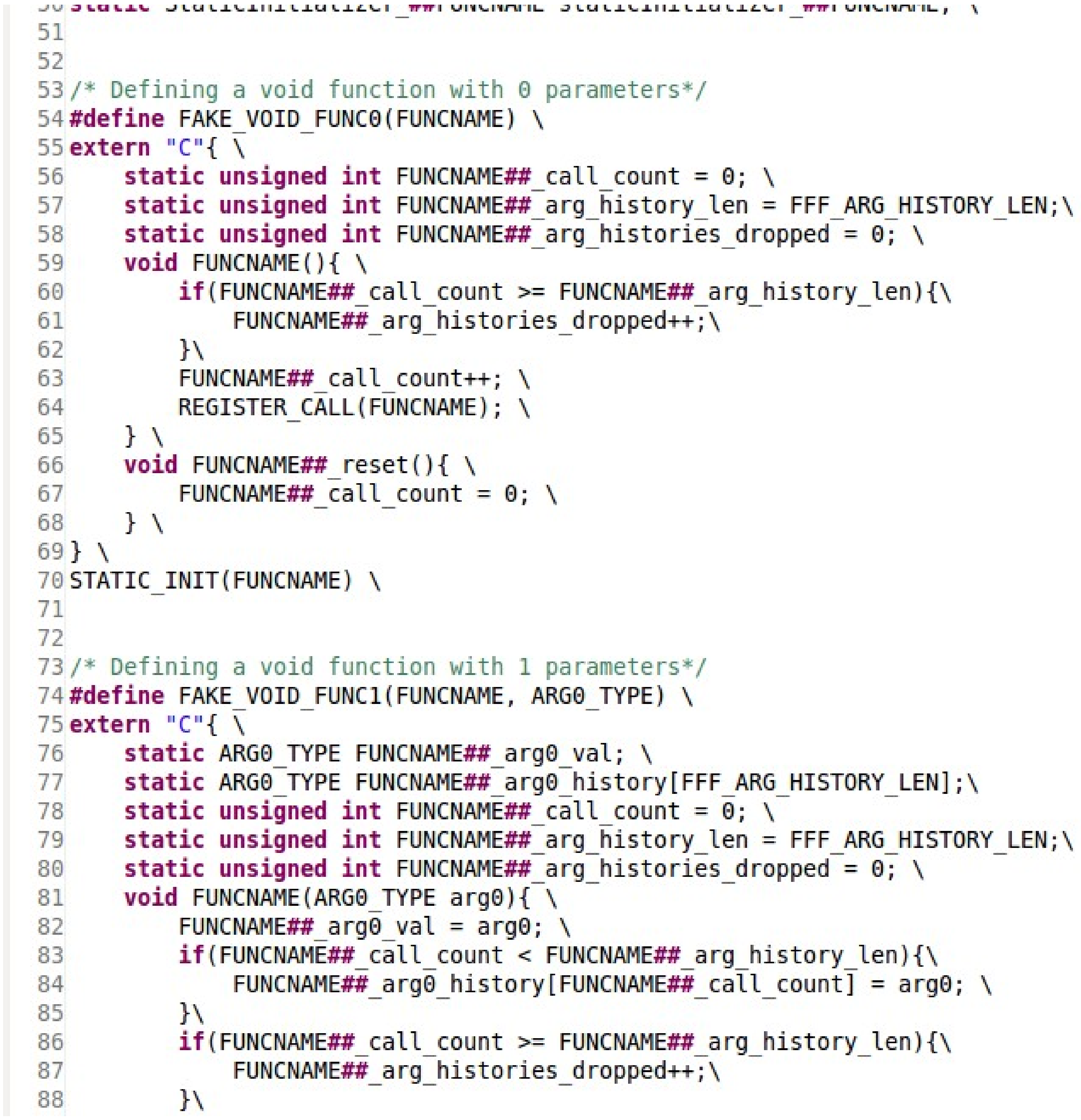

```
TEST_F(FFFTestSuite, return_value_sequences_not_exhausted)
{
    long myReturnVals[3] = { 3, 7, 9 };
    SET_RETURN_SEQ(longfunc0, myReturnVals, 3);
    ASSERT_EQ(myReturnVals[0], longfunc0());
    ASSERT_EQ(myReturnVals[1], longfunc0());
    ASSERT_EQ(myReturnVals[2], longfunc0());
}
```

Capture Argument History!

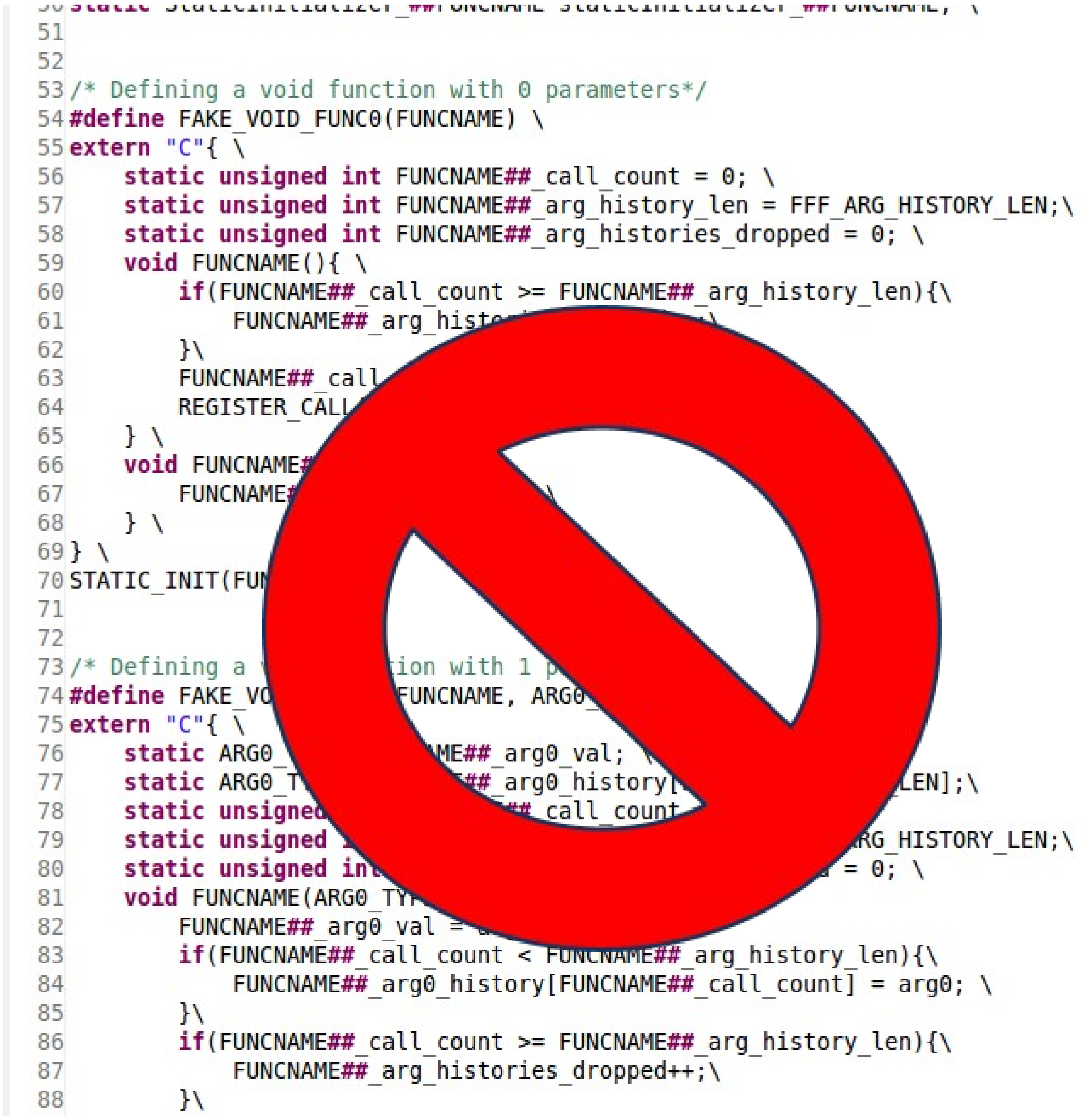


```
TEST_F(FFFTestSuite, when_fake_func_called_then_
{
    voidfunc2('g', 'h');
    ASSERT_EQ('g', voidfunc2_arg0_history[0]);
    ASSERT_EQ('h', voidfunc2_arg1_history[0]);
}
```

how does this work?



```
static StaticInitializer_##FUNCNAME staticInitializer_##FUNCNAME; \


/* Defining a void function with 0 parameters*/
#define FAKE_VOID_FUNC0(FUNCNAME) \
extern "C"{ \
    static unsigned int FUNCNAME##_call_count = 0; \
    static unsigned int FUNCNAME##_arg_history_len = FFF_ARG_HISTORY_LEN;\
    static unsigned int FUNCNAME##_arg_histories_dropped = 0; \
    void FUNCNAME(){ \
        if(FUNCNAME##_call_count >= FUNCNAME##_arg_history_len){\
            FUNCNAME##_arg_histories_dropped++;\
        }\
        FUNCNAME##_call_count++; \
        REGISTER_CALL(FUNCNAME); \
    } \
    void FUNCNAME##_reset(){ \
        FUNCNAME##_call_count = 0; \
    } \
} \
STATIC_INIT(FUNCNAME) \


/* Defining a void function with 1 parameters*/
#define FAKE_VOID_FUNC1(FUNCNAME, ARG0_TYPE) \
extern "C"{ \
    static ARG0_TYPE FUNCNAME##_arg0_val; \
    static ARG0_TYPE FUNCNAME##_arg0_history[FFF_ARG_HISTORY_LEN];\
    static unsigned int FUNCNAME##_call_count = 0; \
    static unsigned int FUNCNAME##_arg_history_len = FFF_ARG_HISTORY_LEN;\
    static unsigned int FUNCNAME##_arg_histories_dropped = 0; \
    void FUNCNAME(ARG0_TYPE arg0){ \
        FUNCNAME##_arg0_val = arg0; \
        if(FUNCNAME##_call_count < FUNCNAME##_arg_history_len){\
            FUNCNAME##_arg0_history[FUNCNAME##_call_count] = arg0; \
        }\
        if(FUNCNAME##_call_count >= FUNCNAME##_arg_history_len){\
            FUNCNAME##_arg_histories_dropped++;\
        }\
```



```
static StaticInitializer_##FUNCNAME staticInitializer_##FUNCNAME; \


/* Defining a void function with 0 parameters*/
#define FAKE_VOID_FUNC0(FUNCNAME) \
extern "C"{ \
    static unsigned int FUNCNAME##_call_count = 0; \
    static unsigned int FUNCNAME##_arg_history_len = FFF_ARG_HISTORY_LEN;\
    static unsigned int FUNCNAME##_arg_histories_dropped = 0; \
    void FUNCNAME(){ \
        if(FUNCNAME##_call_count >= FUNCNAME##_arg_history_len){\
            FUNCNAME##_arg_histo                  \
        }\
        FUNCNAME##_call
        REGISTER_CALL
    } \
    void FUNCNAME#
        FUNCNAME#                    \
    } \
} \
STATIC_INIT(FU


/* Defining a v          tion with 1 p
#define FAKE_VO          FUNCNAME, ARG0_
extern "C"{ \
    static ARG0_          ME##_arg0_val; \
    static ARG0_T          ##_arg0_history[                 LEN];\
    static unsigned          ##_call_count
    static unsigned                                    RG_HISTORY_LEN;\
    static unsigned int                                d = 0; \
    void FUNCNAME(ARG0_TY
        FUNCNAME##_arg0_val = a
        if(FUNCNAME##_call_count < FUNCNAME##_arg_history_len){\
            FUNCNAME##_arg0_history[FUNCNAME##_call_count] = arg0; \
        }\
        if(FUNCNAME##_call_count >= FUNCNAME##_arg_history_len){\
            FUNCNAME##_arg_histories_dropped++;\
        }\
#include "fff.h"
```

#include "fff.h"

https://github.com/meekrosoft/fff

Lightning Talks

Kevlin Henney – Not So SOLID Crew

Mike Long – Unit Testing Legacy C

Didier Verna – Letting Go Of Control (part 2)

Paul Black – Thimama Merodia: A New Authentication Mechanism

Tom Gilb - Simplicity

Matty Williams – I'm Not A Doctor, Trust Me

Didier Verna - (untitled)

Charles Bailey – Massaging Hunks

Seb Rose – Referential Integrity

Richard Harris – Comparing Floats

Jim Hague – Divine Guidance

Bjorn Eriksson – Some Things We Learned...

Phil Nash – CATCH

Letting Go of Control

Part 2/2 Season Finale

Didier Verna

Previously in: “Letting go of control”. . .

- Our software was *out of control*
- This was only going to get *worse*
- We were *afraid*
- We were *ashamed*

But today, everything's going to be fine

- Our software behave like biological realms
- Biological organisms are autonomous

But today, everything's going to be fine

- Our software behave like biological realms
- Biological organisms are autonomous

Our software needs to be autonomous!

- We should not control it
- We should give it the means to control itself instead

What you don't know about GC

The recursive definition of differentiation made no provision for erasure of abandoned list structure. No solution was apparent at the time, but the idea of complicating the elegant definition of differentiation with explicit erasure was unattractive.

– John McCarthy (1978)

The rules of good parenthood

1. **Don’t be too possessive**
   Don’t control everything
2. **Let the teenage crisis pass**
   Bugs and errors are good!
3. **Help your software grow an adult**
   Give it free will
4. **Use Lisp**

The rules of good parenthood

1. **Don’t be too possessive**
   Don’t control everything
2. **Let the teenage crisis pass**
   Bugs and errors are good!
3. **Help your software grow an adult**
   Give it free will
4. **Use Lisp**

The rules of good parenthood

1. **Don’t be too possessive**
   Don’t control everything
2. **Let the teenage crisis pass**
   Bugs and errors are good!
3. **Help your software grow an adult**
   Give it free will
4. **Use Lisp**

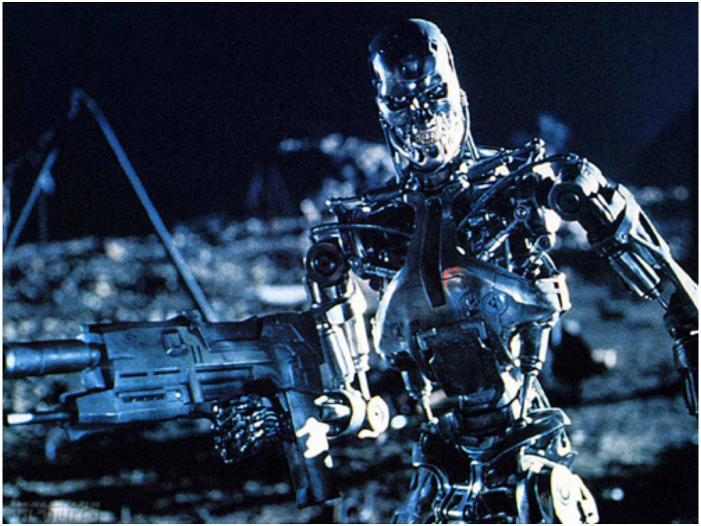
The rules of good parenthood

1. **Don’t be too possessive**
   Don’t control everything
2. **Let the teenage crisis pass**
   Bugs and errors are good!
3. **Help your software grow an adult**
   Give it free will
4. **Use Lisp**

The rules of good parenthood

1. **Don’t be too possessive**
   Don’t control everything
2. **Let the teenage crisis pass**
   Bugs and errors are good!
3. **Help your software grow an adult**
   Give it free will
4. **Use Lisp**

We would not be *afraid* anymore. . .



We would not be *ashamed* anymore. . .

We could be *proud*. . .

Lightning Talks

accu 2011

Kevlin Henney – Not So SOLID Crew

Mike Long – Unit Testing Legacy C

Didier Verna – Letting Go Of Control (part 2)

Paul Black – Thimama Merodia: A New Authentication Mechanism

Tom Gilb - Simplicity

Matty Williams – I'm Not A Doctor, Trust Me

Didier Verna - (untitled)

Charles Bailey – Massaging Hunks

Seb Rose – Referential Integrity

Richard Harris – Comparing Floats

Jim Hague – Divine Guidance

Bjorn Eriksson – Some Things We Learned...

Phil Nash – CATCH

Thimama Merodia: A New Authentication Mechanism

Paul E. Black

drpaule@gmail.com

A lightning talk for ACCU 2011

Identification vs. authentication

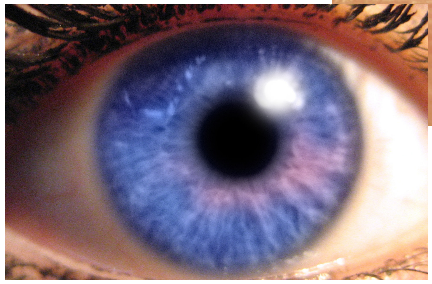
- **Identification answers, who am I?**
  - **Names have limits: many people are named “John Smith”**

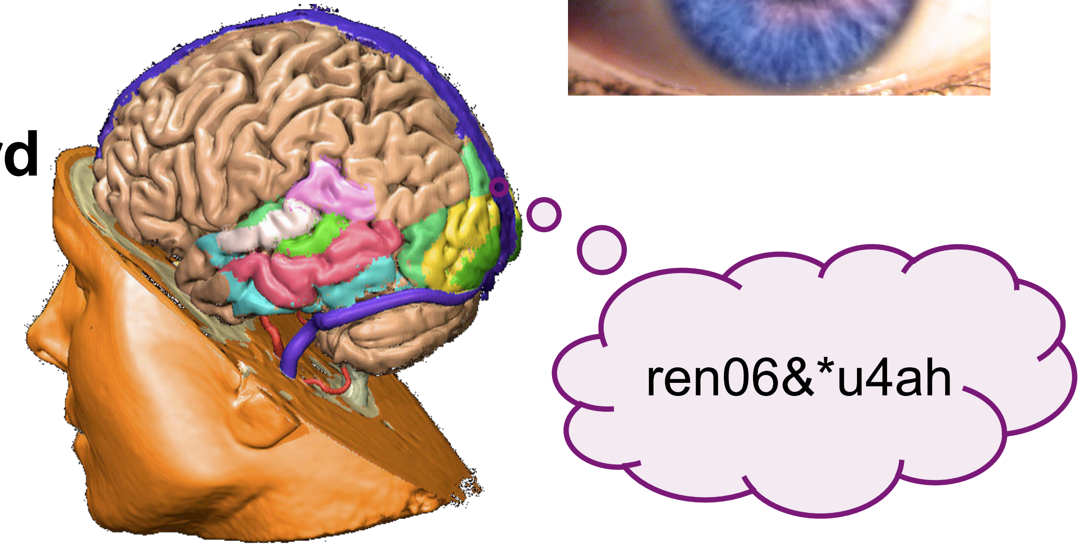
- **Authentication is proof**
  - **Not a subset of identification: possessing a key “proves” I am authorized to open a door**

Authentication Scheme Groups

What you …

- **Have**
  - **token**
- **Are**
  - **physical characteristics**
- **Know**
  - **Password**
- **Can do**

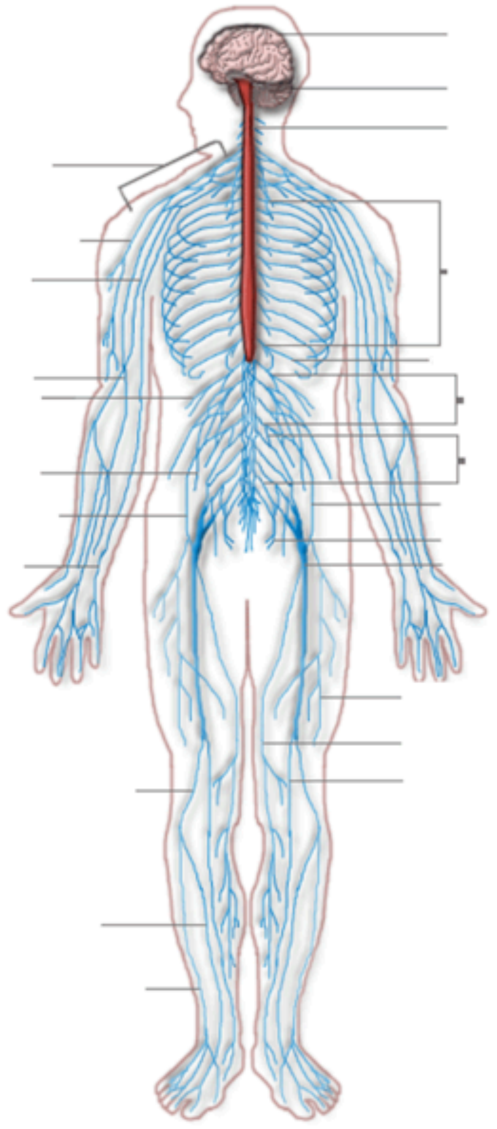




What you can do

- **Computer vision lets us use the other 95% of the neuromuscular system for authentication.**

Gmail Actions
Reply
Reply all
Forward
Send



- **But this is just a tangent …**

Thimama merodia

- **Authenticate by identifying scents made by "stinkjet"**

- **Smells are powerful memory triggers**
- **Humans distinguish 10,000 odors**
- **Hard to describe smells to others**

- ***Open your secret scent envelope and memorize it!***

Lightning Talks

Kevlin Henney – Not So SOLID Crew

Mike Long – Unit Testing Legacy C

Didier Verna – Letting Go Of Control (part 2)

Paul Black – Thimama Merodia: A New Authentication Mechanism

Tom Gilb - Simplicity

Matty Williams – I'm Not A Doctor, Trust Me

Didier Verna - (untitled)

Charles Bailey – Massaging Hunks

Seb Rose – Referential Integrity

Richard Harris – Comparing Floats

Jim Hague – Divine Guidance

Bjorn Eriksson – Some Things We Learned...

Phil Nash – CATCH

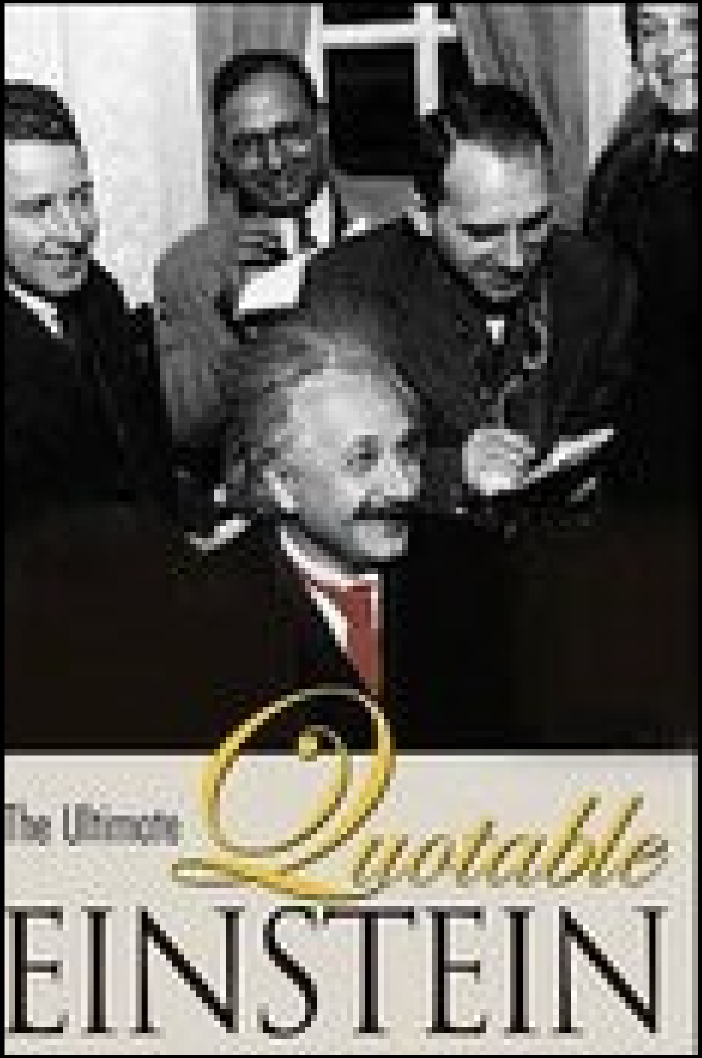
Simplicity

Tom Gilb

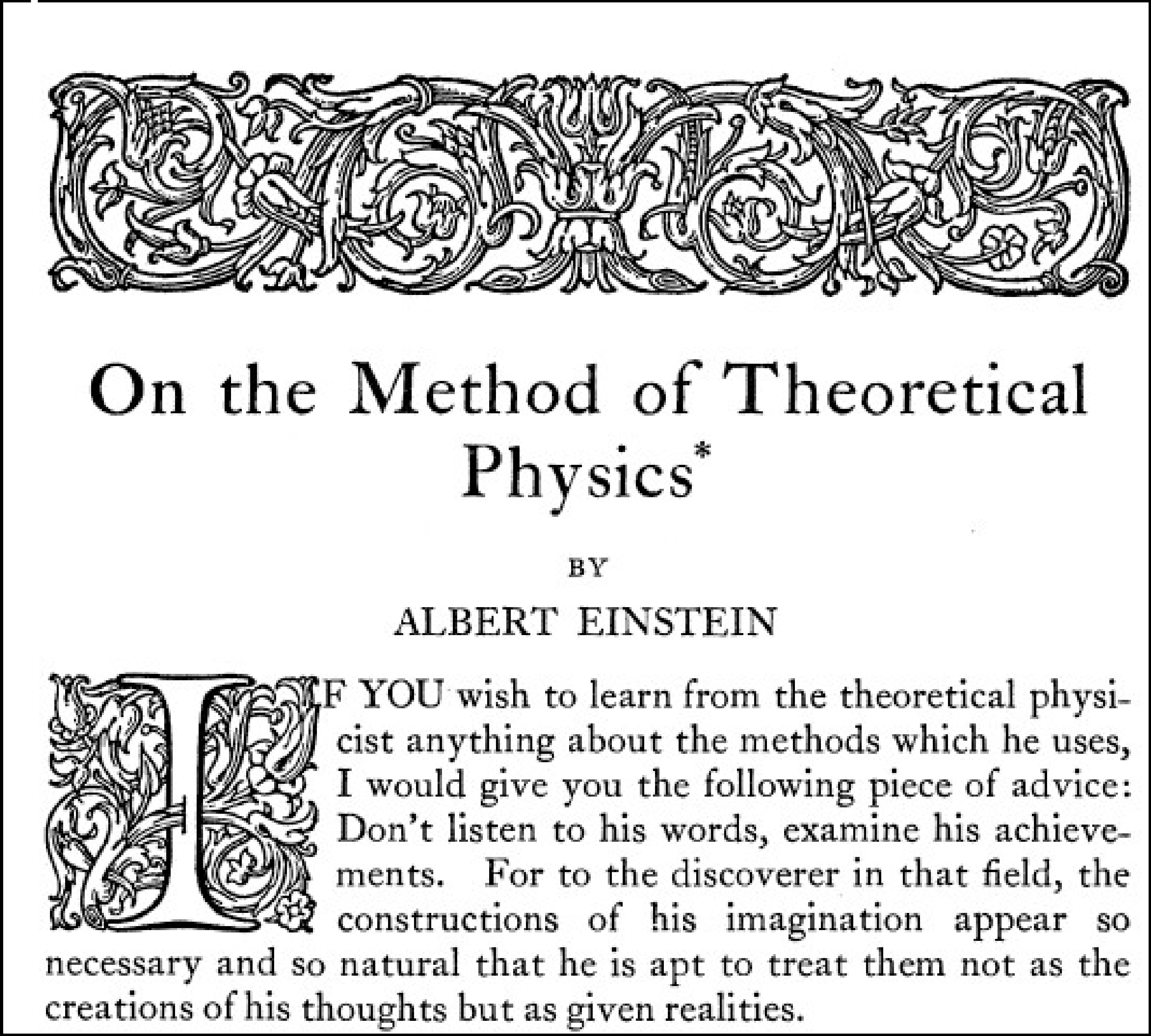
ACCU 5 min Lightening Talk 15 April 2011

How Far should we go?

- "Everything should be made **as simple as possible**, but no **simpler**."
- Attributed to A E but not verified



Spencer 1933 LectureMethod



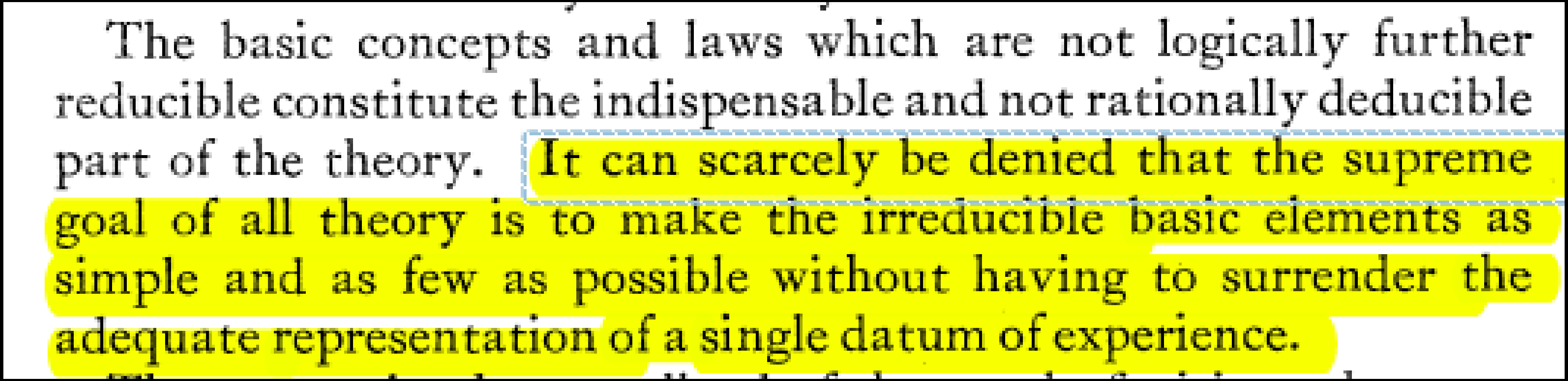
On the Method of Theoretical Physics*

BY

ALBERT EINSTEIN

IF YOU wish to learn from the theoretical physicist anything about the methods which he uses, I would give you the following piece of advice: Don't listen to his words, examine his achievements. For to the discoverer in that field, the constructions of his imagination appear so necessary and so natural that he is apt to treat them not as the creations of his thoughts but as given realities.

What He Really Said!



The basic concepts and laws which are not logically further reducible constitute the indispensable and not rationally deducible part of the theory. It can scarcely be denied that the supreme goal of all theory is to make the irreducible basic elements as simple and as few as possible without having to surrender the adequate representation of a single datum of experience.

Simple Simplicity Principles

1. Simplicity is a defined view of a complex system
2. Simplicity can be measured in several ways
3. You can design systems to be simple to quantified degrees, of defined kinds of simplicity
4. The cost of making a system simple from one stakeholder viewpoint, might be extreme complexity from another viewpoint
5. Complexity might be a price you pay for simplicity elsewhere

Simple Simplicity Principles 2

- 6. The simplest simplicity principle is that simplicity might be complex
- 7. Simplicity is a design tactic to achieve some other aim (such as usability, potability, maintainability)
- 8. You cannot know how how simple your design must be until you know the required level of system qualities, that are your ‘simplicity drivers’
- 9. The cost of simplicity is the cost of the design for simplicity
- 10. Simplicity might cost more than the benefits of the objectives you have stated as requirements.

The Unifying Simplicity Principle

- Simplicity might be complex

Lightning Talks

Kevlin Henney – Not So SOLID Crew

Mike Long – Unit Testing Legacy C

Didier Verna – Letting Go Of Control (part 2)

Paul Black – Thimama Merodia: A New Authentication Mechanism

Tom Gilb - Simplicity

Matty Williams – I'm Not A Doctor, Trust Me

Didier Verna - (untitled)

Charles Bailey – Massaging Hunks

Seb Rose – Referential Integrity

Richard Harris – Comparing Floats

Jim Hague – Divine Guidance

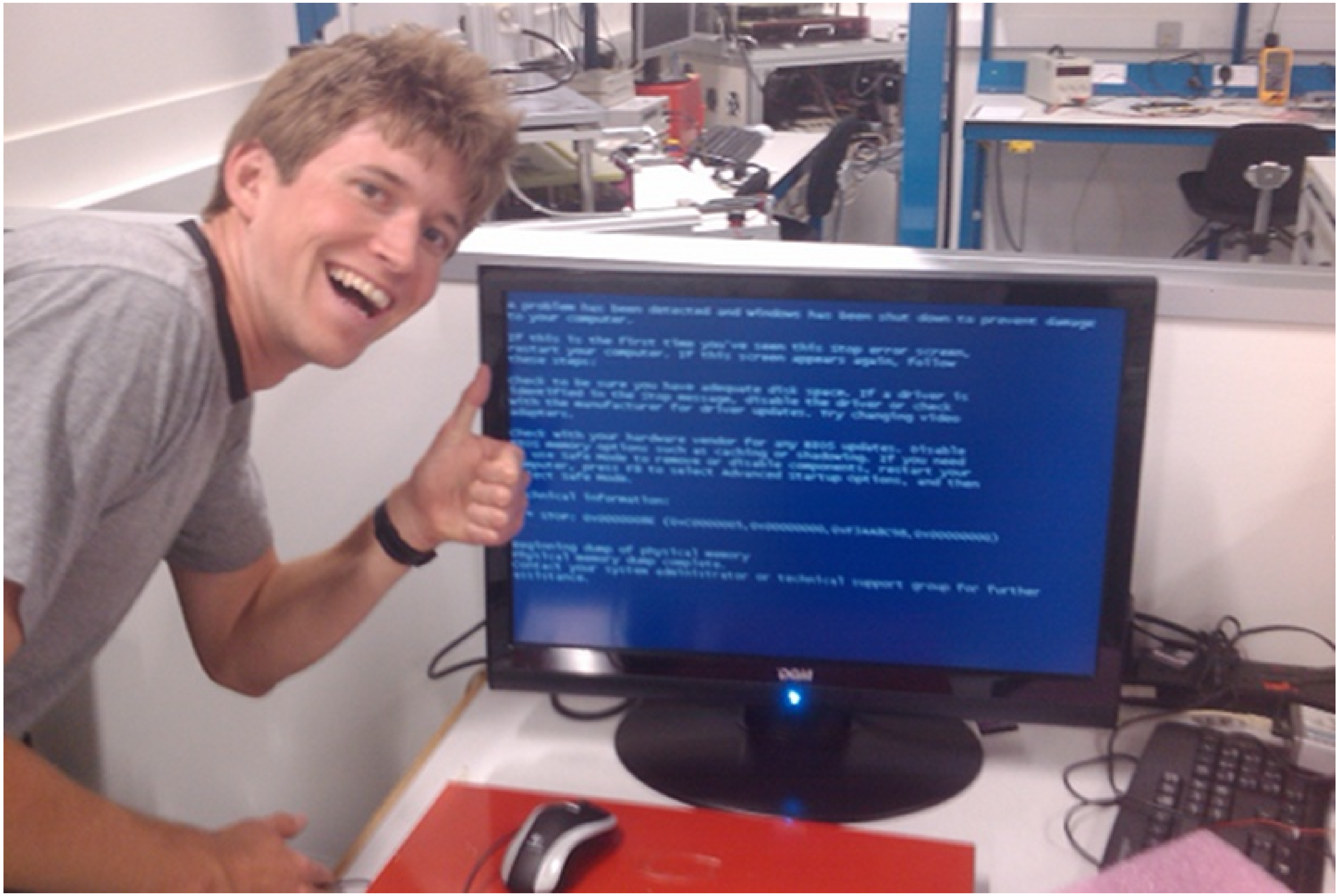
Bjorn Eriksson – Some Things We Learned...

Phil Nash – CATCH

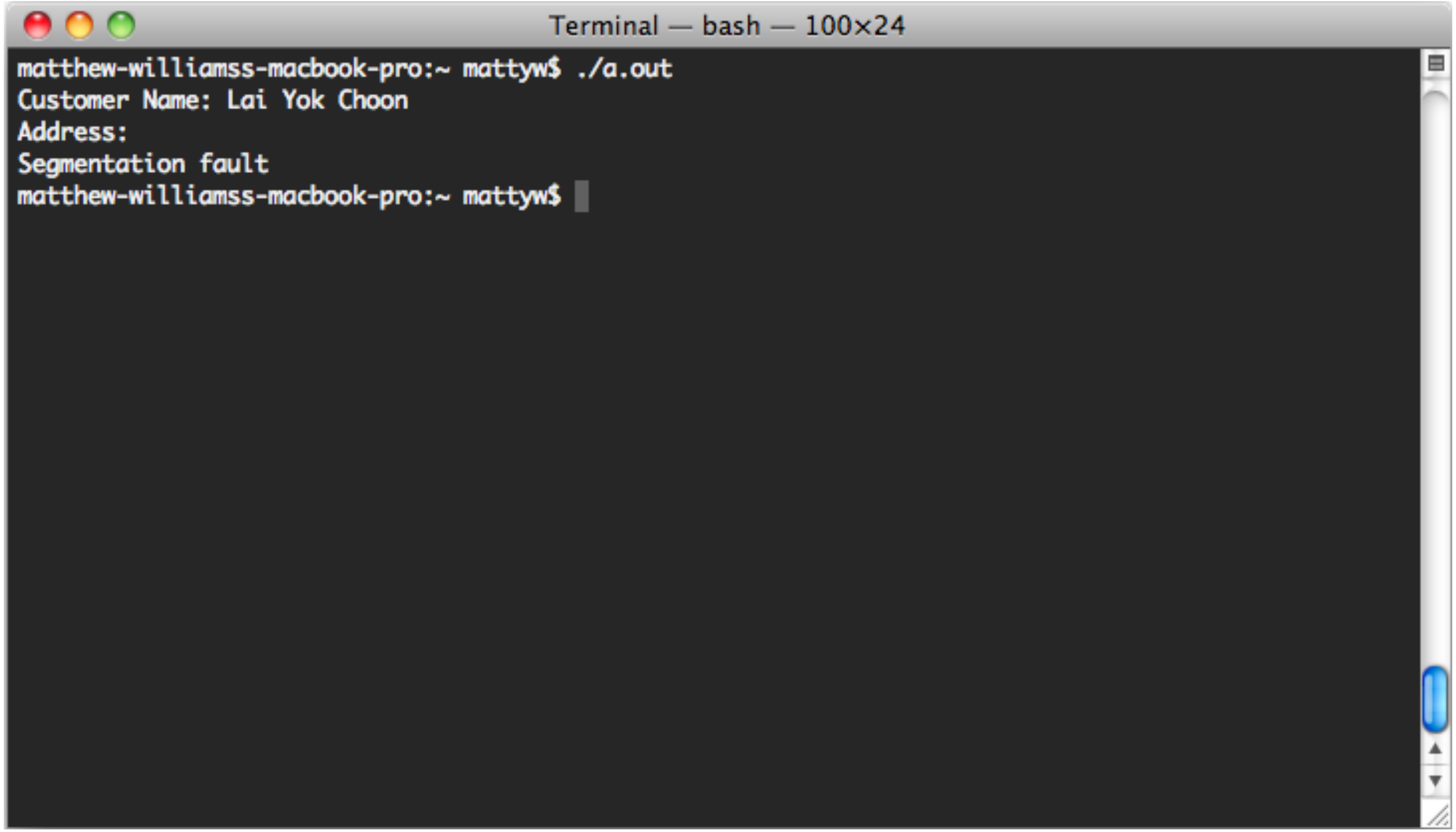
I'm not a doctor, trust me

@matty_jwilliams

CUSTOMER CARE
If We Really Cared for the Customer,
We'd Send Them Somewhere Better.
www.despair.com



```
A problem has been detected and windows has been shut down to prevent damage
to your computer.

If this is the first time you've seen this Stop error screen,
restart your computer. If this screen appears again, follow
these steps:

Check to be sure you have adequate disk space. If a driver is
identified in the Stop message, disable the driver or check
with the manufacturer for driver updates. Try changing video
adapters.

Check with your hardware vendor for any BIOS updates. Disable
BIOS memory options such as caching or shadowing. If you need
 use Safe Mode to remove or disable components, restart your
mputer, press F8 to select Advanced Startup Options, and then
lect Safe Mode.

chnical information:

* STOP: 0x0000008E (0xC0000005,0x00000000,0xF1AADC98,0x00000000)

Beginning dump of physical memory
Physical memory dump complete.
Contact your system administrator or technical support group for further
assistance.
```



```
matthew-williamss-macbook-pro:~ mattyw$ ./a.out
Customer Name: Lai Yok Choon
Address:
Segmentation fault
matthew-williamss-macbook-pro:~ mattyw$ 
```

feedback?

Codes of Conduct

- GMC Duties of a Doctor
- HPC Standards of Practice

My Code of Conduct

- Make care of customers my first concern.
- Protect and promote the health of my code.
- Work in partnership with customers.
- Keep my knowledge and skills up to date.

Lightning Talks

Kevlin Henney – Not So SOLID Crew

Mike Long – Unit Testing Legacy C

Didier Verna – Letting Go Of Control (part 2)

Paul Black – Thimama Merodia: A New Authentication Mechanism

Tom Gilb - Simplicity

Matty Williams – I'm Not A Doctor, Trust Me

Didier Verna - (untitled)

Charles Bailey – Massaging Hunks

Seb Rose – Referential Integrity

Richard Harris – Comparing Floats

Jim Hague – Divine Guidance

Bjorn Eriksson – Some Things We Learned...

Phil Nash – CATCH

@didierverna

Lightning Talks

Kevlin Henney – Not So SOLID Crew

Mike Long – Unit Testing Legacy C

Didier Verna – Letting Go Of Control (part 2)

Paul Black – Thimama Merodia: A New Authentication Mechanism

Tom Gilb - Simplicity

Matty Williams – I'm Not A Doctor, Trust Me

Didier Verna - (untitled)

Charles Bailey – Massaging Hunks

Seb Rose – Referential Integrity

Richard Harris – Comparing Floats

Jim Hague – Divine Guidance

Bjorn Eriksson – Some Things We Learned...

Phil Nash – CATCH

Massaging Hunks

The Awesome Power of `git add -p`

Charles Bailey

Igence Ltd

15th April 2011

The Cache

- git has a staging area

The Cache

- git has a staging area
- a.k.a. index

The Cache

- `git` has a staging area
- a.k.a. index
- a.k.a. cache

What does it allow?

The indexcachestagingarea allows you to easily maintain a “dirty” working tree, enabling you to commit just the changes that you want to commit.

What does it allow?

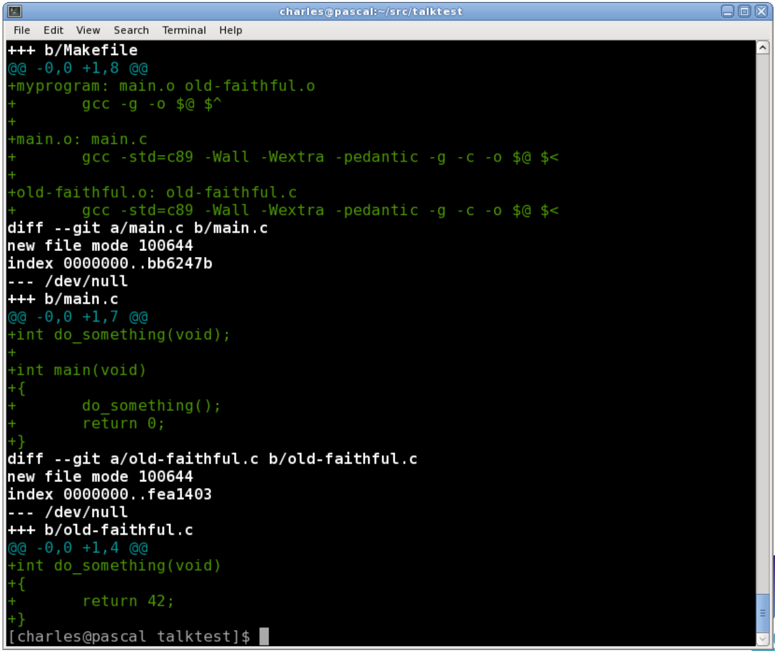
The indexcachestagingarea allows you to *relatively* easily maintain a “dirty” working tree, enabling you to commit just the changes that you want to commit.

`git add -p`

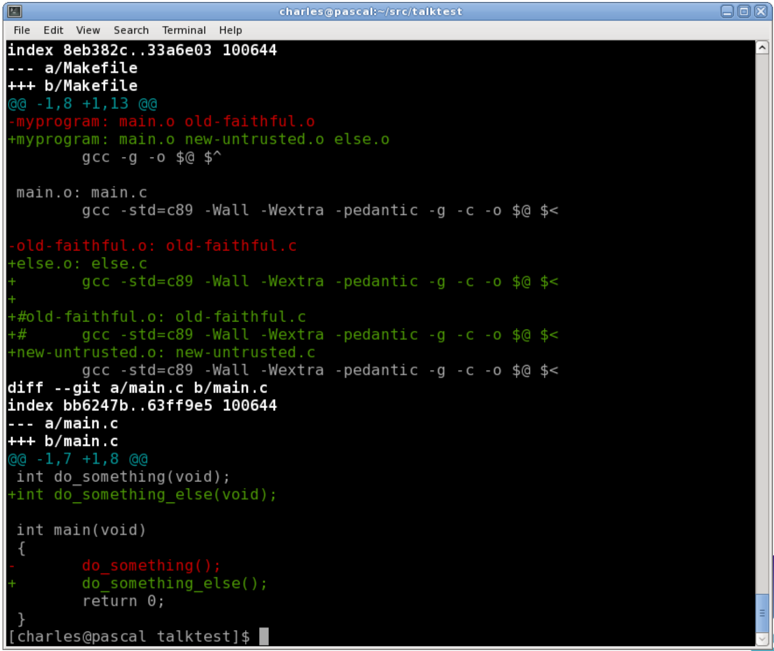
- `git add` allows you to stage only some files.
- `git add -p` allows you to stage only some non-overlapping hunks from the same file.
- `git add -p` with “`edit`” allows you to stage anything

git add -p

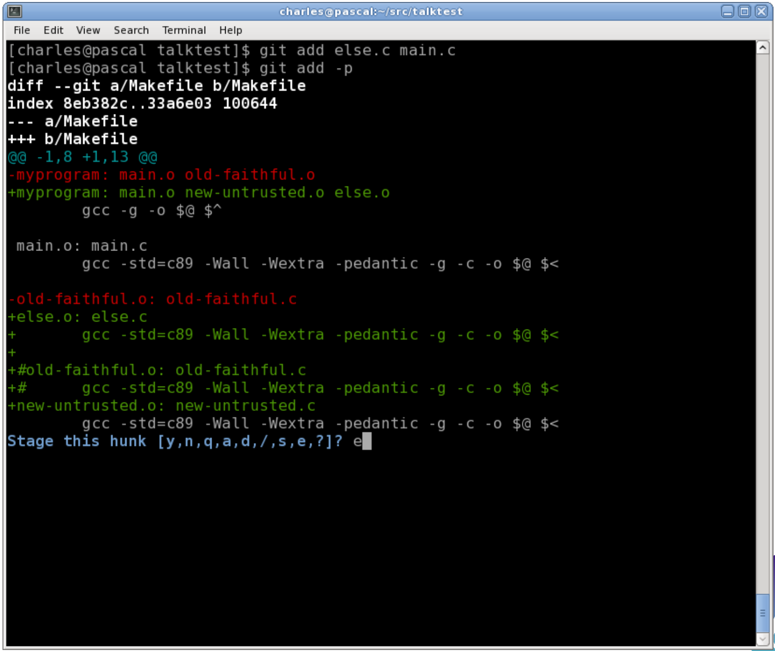
- git add allows you to stage only some files.
- git add -p allows you to stage only some non-overlapping hunks from the same file.
- git add -p with "edit" allows you to stage anything
  . . . including things that aren't even in your working tree.


charles@pascal:~/src/talktest
File Edit View Search Terminal Help

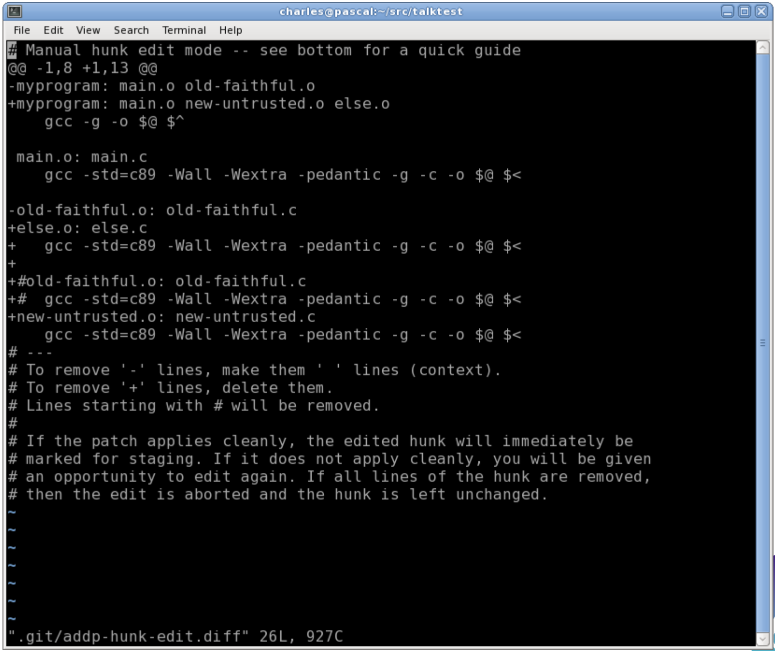
```
+++ b/Makefile
@@ -0,0 +1,8 @@
+myprogram: main.o old-faithful.o
+       gcc -g -o $@ $^
+
+main.o: main.c
+       gcc -std=c89 -Wall -Wextra -pedantic -g -c -o $@ $<
+
+old-faithful.o: old-faithful.c
+       gcc -std=c89 -Wall -Wextra -pedantic -g -c -o $@ $<
diff --git a/main.c b/main.c
new file mode 100644
index 0000000..bb6247b
--- /dev/null
+++ b/main.c
@@ -0,0 +1,7 @@
+int do_something(void);
+
+int main(void)
+{
+       do_something();
+       return 0;
+}
diff --git a/old-faithful.c b/old-faithful.c
new file mode 100644
index 0000000..fea1403
--- /dev/null
+++ b/old-faithful.c
@@ -0,0 +1,4 @@
+int do_something(void)
+{
+       return 42;
+}
[charles@pascal talktest]$
```


charles@pascal:~/src/talktest
File Edit View Search Terminal Help

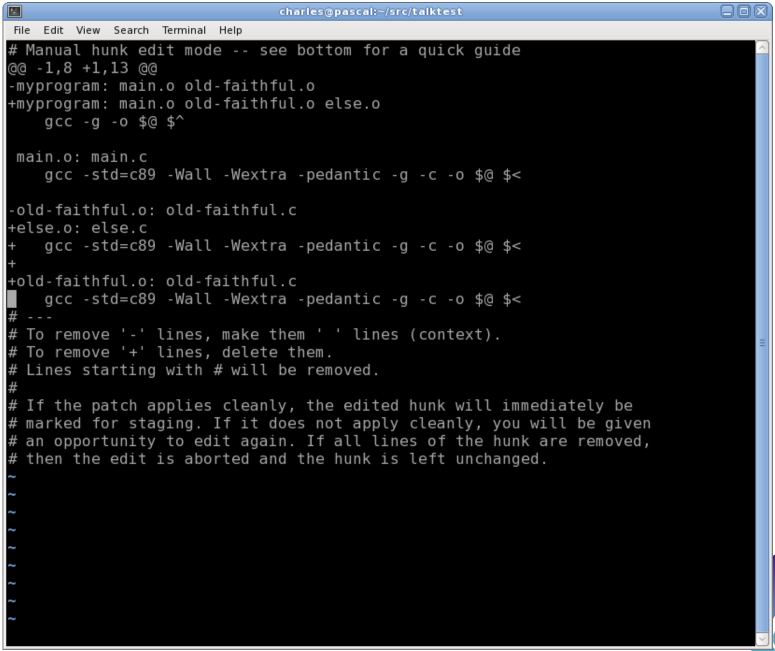
```
index 8eb382c..33a6e03 100644
--- a/Makefile
+++ b/Makefile
@@ -1,8 +1,13 @@
-myprogram: main.o old-faithful.o
+myprogram: main.o new-untrusted.o else.o
        gcc -g -o $@ $^

 main.o: main.c
        gcc -std=c89 -Wall -Wextra -pedantic -g -c -o $@ $<

-old-faithful.o: old-faithful.c
+else.o: else.c
+       gcc -std=c89 -Wall -Wextra -pedantic -g -c -o $@ $<
+
+#old-faithful.o: old-faithful.c
+#      gcc -std=c89 -Wall -Wextra -pedantic -g -c -o $@ $<
+new-untrusted.o: new-untrusted.c
        gcc -std=c89 -Wall -Wextra -pedantic -g -c -o $@ $<
diff --git a/main.c b/main.c
index bb6247b..63ff9e5 100644
--- a/main.c
+++ b/main.c
@@ -1,7 +1,8 @@
 int do_something(void);
+int do_something_else(void);

 int main(void)
 {
-       do_something();
+       do_something_else();
        return 0;
 }
[charles@pascal talktest]$
```


charles@pascal:~/src/talktest
File Edit View Search Terminal Help

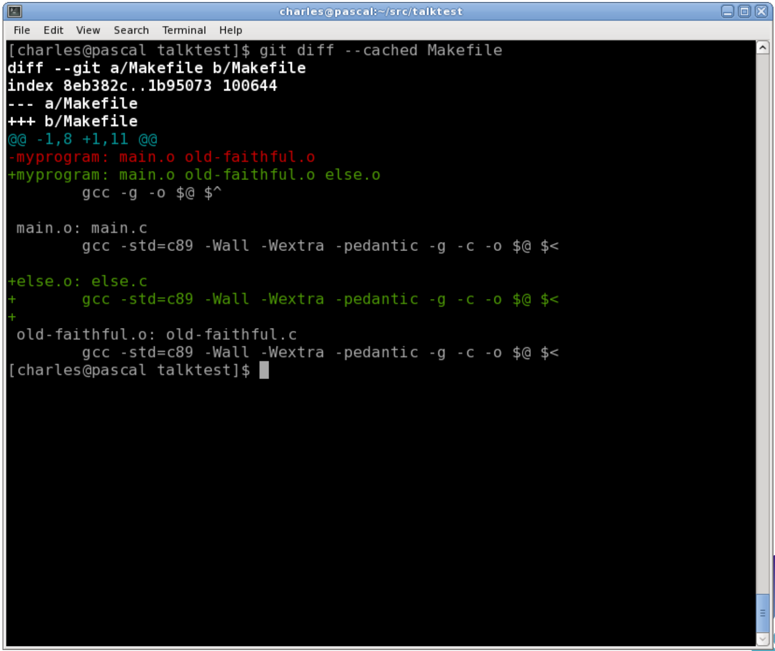
```
[charles@pascal talktest]$ git add else.c main.c
[charles@pascal talktest]$ git add -p
diff --git a/Makefile b/Makefile
index 8eb382c..33a6e03 100644
--- a/Makefile
+++ b/Makefile
@@ -1,8 +1,13 @@
-myprogram: main.o old-faithful.o
+myprogram: main.o new-untrusted.o else.o
        gcc -g -o $@ $^

 main.o: main.c
        gcc -std=c89 -Wall -Wextra -pedantic -g -c -o $@ $<

-old-faithful.o: old-faithful.c
+else.o: else.c
+       gcc -std=c89 -Wall -Wextra -pedantic -g -c -o $@ $<
+
+#old-faithful.o: old-faithful.c
+#      gcc -std=c89 -Wall -Wextra -pedantic -g -c -o $@ $<
+new-untrusted.o: new-untrusted.c
        gcc -std=c89 -Wall -Wextra -pedantic -g -c -o $@ $<
Stage this hunk [y,n,q,a,d,/,s,e,?]? e
```


charles@pascal:~/src/talktest
File Edit View Search Terminal Help

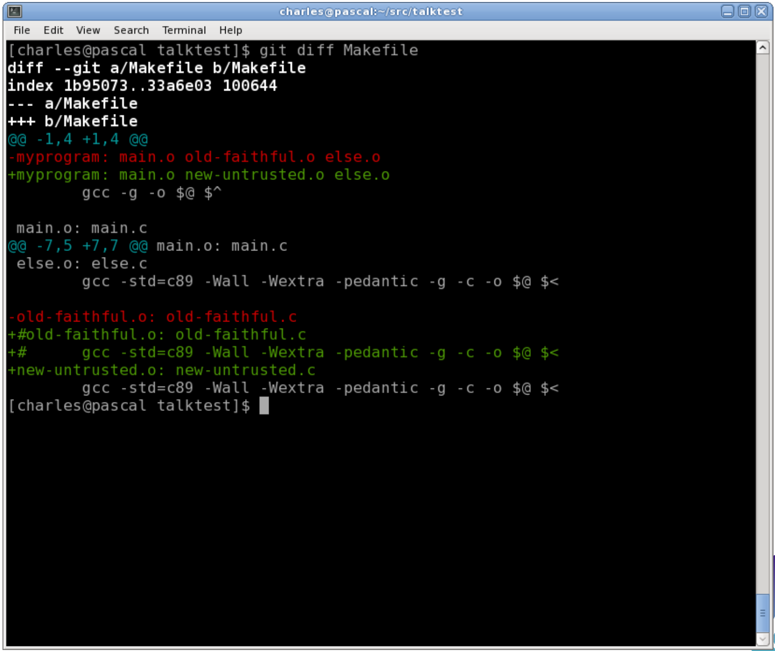
```
# Manual hunk edit mode -- see bottom for a quick guide
@@ -1,8 +1,13 @@
-myprogram: main.o old-faithful.o
+myprogram: main.o new-untrusted.o else.o
    gcc -g -o $@ $^

 main.o: main.c
    gcc -std=c89 -Wall -Wextra -pedantic -g -c -o $@ $<

-old-faithful.o: old-faithful.c
+else.o: else.c
+   gcc -std=c89 -Wall -Wextra -pedantic -g -c -o $@ $<
+
+#old-faithful.o: old-faithful.c
+#  gcc -std=c89 -Wall -Wextra -pedantic -g -c -o $@ $<
+new-untrusted.o: new-untrusted.c
    gcc -std=c89 -Wall -Wextra -pedantic -g -c -o $@ $<
# ---
# To remove '-' lines, make them ' ' lines (context).
# To remove '+' lines, delete them.
# Lines starting with # will be removed.
#
# If the patch applies cleanly, the edited hunk will immediately be
# marked for staging. If it does not apply cleanly, you will be given
# an opportunity to edit again. If all lines of the hunk are removed,
# then the edit is aborted and the hunk is left unchanged.
~
~
~
~
~
~
~
".git/addp-hunk-edit.diff" 26L, 927C
```


charles@pascal:~/src/talktest
File Edit View Search Terminal Help

```
# Manual hunk edit mode -- see bottom for a quick guide
@@ -1,8 +1,13 @@
-myprogram: main.o old-faithful.o
+myprogram: main.o old-faithful.o else.o
    gcc -g -o $@ $^

 main.o: main.c
    gcc -std=c89 -Wall -Wextra -pedantic -g -c -o $@ $<

-old-faithful.o: old-faithful.c
+else.o: else.c
+   gcc -std=c89 -Wall -Wextra -pedantic -g -c -o $@ $<
+
+old-faithful.o: old-faithful.c
    gcc -std=c89 -Wall -Wextra -pedantic -g -c -o $@ $<
# ---
# To remove '-' lines, make them ' ' lines (context).
# To remove '+' lines, delete them.
# Lines starting with # will be removed.
#
# If the patch applies cleanly, the edited hunk will immediately be
# marked for staging. If it does not apply cleanly, you will be given
# an opportunity to edit again. If all lines of the hunk are removed,
# then the edit is aborted and the hunk is left unchanged.
~
~
~
~
~
~
~
~
~
```


charles@pascal:~/src/talktest
File Edit View Search Terminal Help

```
[charles@pascal talktest]$ git diff --cached Makefile
diff --git a/Makefile b/Makefile
index 8eb382c..1b95073 100644
--- a/Makefile
+++ b/Makefile
@@ -1,8 +1,11 @@
-myprogram: main.o old-faithful.o
+myprogram: main.o old-faithful.o else.o
        gcc -g -o $@ $^

 main.o: main.c
        gcc -std=c89 -Wall -Wextra -pedantic -g -c -o $@ $<

+else.o: else.c
+       gcc -std=c89 -Wall -Wextra -pedantic -g -c -o $@ $<
+
 old-faithful.o: old-faithful.c
        gcc -std=c89 -Wall -Wextra -pedantic -g -c -o $@ $<
[charles@pascal talktest]$ 
```


charles@pascal:~/src/talktest
File Edit View Search Terminal Help

```
[charles@pascal talktest]$ git diff Makefile
diff --git a/Makefile b/Makefile
index 1b95073..33a6e03 100644
--- a/Makefile
+++ b/Makefile
@@ -1,4 +1,4 @@
-myprogram: main.o old-faithful.o else.o
+myprogram: main.o new-untrusted.o else.o
        gcc -g -o $@ $^

 main.o: main.c
@@ -7,5 +7,7 @@ main.o: main.c
 else.o: else.c
        gcc -std=c89 -Wall -Wextra -pedantic -g -c -o $@ $<

-old-faithful.o: old-faithful.c
+#old-faithful.o: old-faithful.c
+#      gcc -std=c89 -Wall -Wextra -pedantic -g -c -o $@ $<
+new-untrusted.o: new-untrusted.c
        gcc -std=c89 -Wall -Wextra -pedantic -g -c -o $@ $<
[charles@pascal talktest]$
```

Avoiding the working tree

A simple example

Not the only way to avoid using the working tree

```
git update-index --cacheinfo 100644 $(
        git show :Makefile |
        sed -e's/old-faithful/new-untrusted/g' |
        git hash-object -w --stdin ) Makefile
```

Not the only way to avoid using the working tree

```
git update-index --cacheinfo 100644 $(
        git show :Makefile |
        sed -e's/old-faithful/new-untrusted/g' |
        git hash-object -w --stdin ) Makefile
```

Not the only way to avoid using the working tree

```
git update-index --cacheinfo 100644 $(
        git show :Makefile |
        sed -e's/old-faithful/new-untrusted/g' |
        git hash-object -w --stdin ) Makefile
```

Not the only way to avoid using the working tree

```
git update-index --cacheinfo 100644 $(
        git show :Makefile |
        sed -e's/old-faithful/new-untrusted/g' |
        git hash-object -w --stdin ) Makefile
```

Not the only way to avoid using the working tree

```
git update-index --cacheinfo 100644 $(
        git show :Makefile |
        sed -e's/old-faithful/new-untrusted/g' |
        git hash-object -w --stdin ) Makefile
```

Not the only way to avoid using the working tree

```
git update-index --cacheinfo 100644 $(
        git show :Makefile |
        sed -e's/old-faithful/new-untrusted/g' |
        git hash-object -w --stdin ) Makefile
```

A slide without a really ugly shell pipeline

Warning: irresponsible use of git can seriously damage your sanity.

Lightning Talks

Kevlin Henney – Not So SOLID Crew

Mike Long – Unit Testing Legacy C

Didier Verna – Letting Go Of Control (part 2)

Paul Black – Thimama Merodia: A New Authentication Mechanism

Tom Gilb - Simplicity

Matty Williams – I'm Not A Doctor, Trust Me

Didier Verna - (untitled)

Charles Bailey – Massaging Hunks

Seb Rose – Referential Integrity

Richard Harris – Comparing Floats

Jim Hague – Divine Guidance

Bjorn Eriksson – Some Things We Learned...

Phil Nash – CATCH

Lightning Talks

Kevlin Henney – Not So SOLID Crew

Mike Long – Unit Testing Legacy C

Didier Verna – Letting Go Of Control (part 2)

Paul Black – Thimama Merodia: A New Authentication Mechanism

Tom Gilb - Simplicity

Matty Williams – I'm Not A Doctor, Trust Me

Didier Verna - (untitled)

Charles Bailey – Massaging Hunks

Seb Rose – Referential Integrity

Richard Harris – Comparing Floats

Jim Hague – Divine Guidance

Bjorn Eriksson – Some Things We Learned...

Phil Nash – CATCH

Lightning Talks

- Kevlin Henney – Not So SOLID Crew
- Mike Long – Unit Testing Legacy C
- Didier Verna – Letting Go Of Control (part 2)
- Paul Black – Thimama Merodia: A New Authentication Mechanism
- Tom Gilb - Simplicity
- Matty Williams – I'm Not A Doctor, Trust Me
- Didier Verna - (untitled)
- Charles Bailey – Massaging Hunks
- Seb Rose – Referential Integrity
- Richard Harris – Comparing Floats
- **Jim Hague – Divine Guidance**
- **Bjorn Eriksson – Some Things We Learned...**
- **Phil Nash – CATCH**

Divine Guidance

A Lightning Talk

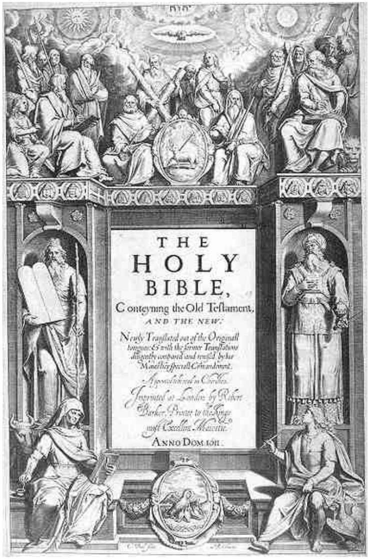
Jim Hague

LAIC Ag

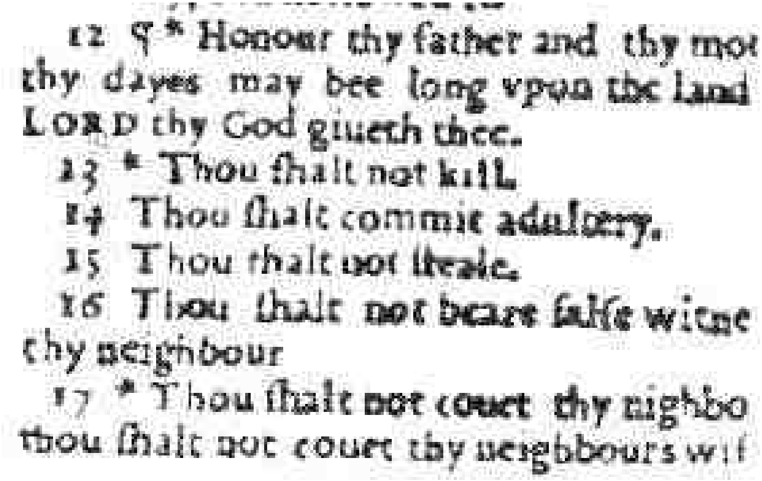
ACCU Conference 2011

Oxford Folk Weekend

http://www.folkweekendoxford.co.uk/



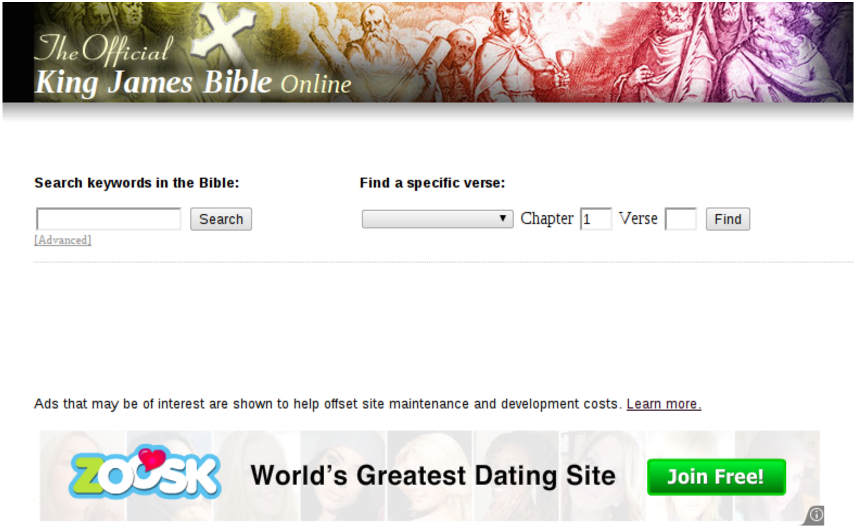
THE
HOLY
BIBLE,
Conteyning the Old Teſtament,
AND THE NEW:
Newly Tranſlated out of the Originall
tongues: & with the former Tranſlations
diligently compared and reuiſed, by his
Maiesties ſpeciall Commandement.
Appointed to be read in Churches.
Imprinted at London by Robert
Barker, Printer to the Kings
moſt Excellent Maieſtie.
ANNO DOM. 1611.



12 ¶ * Honour thy father and thy mot
thy dayes may bee long vpon the land
LORD thy God giueth thee.
13 * Thou ſhalt not kill.
14 Thou ſhalt commit adultery.
15 Thou ſhalt not ſteale.
16 Thou ſhalt not beare falſe witne
thy neighbour
17 * Thou ſhalt not couet thy nighbo
thou ſhalt not couet thy neighbours wif

highways and hedges; clear as crystal; still small voice; hip and thigh; arose as one man; lick the dust; a thorn in the flesh; broken reed; root of all evil; sweat of his brow; heap coals of fire; a law unto themselves; the fat of the land; dark sayings; a soft answer; a word in season; weighed in the balance and found wanting; we are the people; the full measure of justice is not meted out to them; they sold their birthright for a mess of pottage; they have fallen among thieves.

Bible Baptist Church:
We believe The King James Bible is the word of God, and is the final authority in all matters of faith and practice.



Search keywords in the Bible:

Search
[Advanced]

Find a specific verse:

Chapter 1 Verse Find

Ads that may be of interest are shown to help offset site maintenance and development costs. Learn more.

- Static or dynamic typing

- Static or dynamic typing
- Brace position

- Static or dynamic typing
- Brace position
- Procedural or functional

- Static or dynamic typing
- Brace position
- Procedural or functional
- Recurse

- Static or dynamic typing
- Brace position
- Procedural or functional
- Recurse
- Source

- Static or dynamic typing
- Brace position
- Procedural or functional
- Recurse
- Source
- Agile

- Static or dynamic typing
- Brace position
- Procedural or functional
- Recurse
- Source
- Agile
- Scrum

- Static or dynamic typing
- Brace position
- Procedural or functional
- Recurse
- Source
- Agile
- Scrum
- Team

- Memory

- Memory
  Proverbs 10:7

- Memory
  Proverbs 10:7
  The memory of the just [is] blessed: but the name of the wicked shall rot.

- Manage

- Manage
  Additions to Esther 16:5

- Manage

  Additions to Esther 16:5

  Oftentimes also fair speech of those, that are put in trust to manage their friends' affairs, hath caused many that are in authority to be partakers of innocent blood, and hath enwrapped them in remediless calamities:

- Release

- Release
  Deuteronomy 15:1

- Release
  Deuteronomy 15:1
  At the end of [every] seven years thou shalt make a release.

Lightning Talks

Kevlin Henney – Not So SOLID Crew

Mike Long – Unit Testing Legacy C

Didier Verna – Letting Go Of Control (part 2)

Paul Black – Thimama Merodia: A New Authentication Mechanism

Tom Gilb - Simplicity

Matty Williams – I'm Not A Doctor, Trust Me

Didier Verna - (untitled)

Charles Bailey – Massaging Hunks

Seb Rose – Referential Integrity

Richard Harris – Comparing Floats

Jim Hague – Divine Guidance

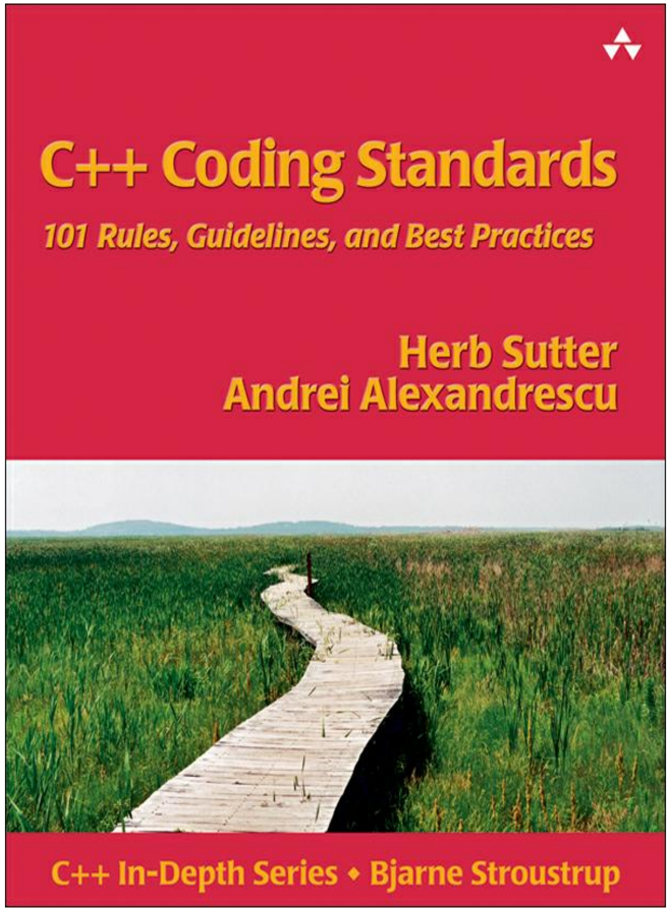
Bjorn Eriksson – Some Things We Learned...

Phil Nash – CATCH

accu 2011

Some things we learned on a greenfield embedded project using modern C++

We had seen the light. and it was good.


C++ Coding Standards
101 Rules, Guidelines, and Best Practices
Herb Sutter
Andrei Alexandrescu
C++ In-Depth Series • Bjarne Stroustrup

Confidentiality Agreement

It is understood and agreed to that the Discloser and the Recipient would like to exchange certain information that may be considered confidential. To ensure the protection of such information and in consideration of the agreement to exchange said information, the parties agree as follows:

1. The confidential information to be disclosed by Discloser under this Agreement (“Confidential Information”) can be described as and includes. In addition to the above, Confidential Information shall also include, and the Recipient shall have a duty to protect, other confidential and/or sensitive information which is (a) disclosed by Discloser in writing and marked as confidential (or with other similar designation) at the time of disclosure; and/or (b) disclosed by Discloser in any other manner and identified as confidential at the time of disclosure and is also summarized and designated as confidential in a written memorandum delivered to Recipient within thirty (30) days of the disclosure.
2. Recipient shall use the Confidential Information only for the purpose of evaluating potential business and investment relationships with Discloser.
3. Recipient shall limit disclosure of Confidential Information within its own organization to its directors, officers, partners, members and/or employees having a need to know and shall not disclose Confidential Information to any third party (whether an individual, corporation, or other entity) without the prior written consent of Discloser. Recipient shall have satisfied its obligations under this paragraph if it takes affirmative measures to ensure compliance with these confidentiality obligations by its employees, agents, consultants and others who are permitted access to or use of the Confidential Information.
4. This Agreement imposes no obligation upon Recipient with respect to any Confidential Information (a) that was in Recipient’s possession before receipt from Discloser; (b) is or becomes a matter of public knowledge through no fault of Recipient; (c) is rightfully received by Recipient from a third party not owing a duty of confidentiality to the Discloser; (d) is disclosed without a duty of confidentiality to a third party by, or with the authorization of, Discloser; or (e) is independently developed by Recipient.
5. Discloser warrants that he/she has the right to make the disclosures under this Agreement.

A couple of highlights

- signals2
- ptr_list, ptr_vec, et al
- shared_ptr<>

```
class collection_of_data
{
   // data  to protect
   // ....

public:
   typedef boost::shared_ptr<dynamic_data> locking_ptr;

   locking_ptr write_access() const
   {
      lock();

      return boost::shared_ptr< collection_of_data >(
         const_cast< collection_of_data *>(this),
         boost::mem_fn(&unlock));
   }

   ... unlock()
};
```

Recommended reading

Scott Meyers on custom deleters

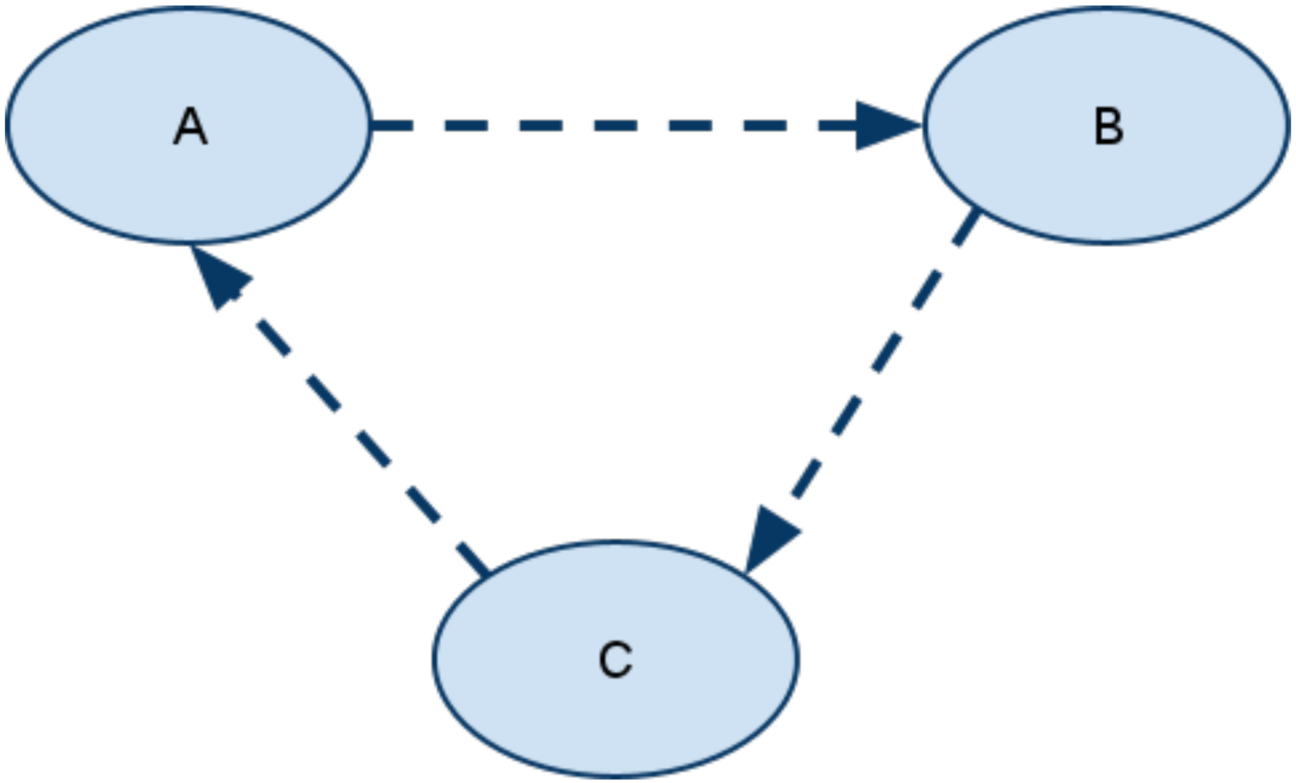
http://www.artima.com/cppsource/top_cpp_aha_moments.html

Smart Pointer Programming Techniques

http://www.boost.org/doc/libs/release/libs/smart_ptr/sp_techniques.html

This left us sleeping well, secure in our knowledge that no memory was leaked nor overwritten.

Or...?



A classic!

Helpful suggestions are worth a drink in the bar

Lightning Talks

Kevlin Henney – Not So SOLID Crew

Mike Long – Unit Testing Legacy C

Didier Verna – Letting Go Of Control (part 2)

Paul Black – Thimama Merodia: A New Authentication Mechanism

Tom Gilb - Simplicity

Matty Williams – I'm Not A Doctor, Trust Me

Didier Verna - (untitled)

Charles Bailey – Massaging Hunks

Seb Rose – Referential Integrity

Richard Harris – Comparing Floats

Jim Hague – Divine Guidance

Bjorn Eriksson – Some Things We Learned...

Phil Nash – CATCH