accu 2013

Lightning Talks

Pete Goodliffe – Design Sins

Alan Griffiths – The ACCU: Magic Happens Here

Dirk Haun – What's our Status?

Gail Ollis – Hello. I'm Back

Michel Grootjans – The Librarian

Björn Fahller – Why Are (only) We Here?

Frank Birbacher – Simple Quick Sort in C++

Calum Grant – C++ Active Objects

Didier Verna – Communities

Burkhard Kloss – Just a Minute

Guy Bolton King – BDD with Boost Test

Ed Sykes – A Decision Made With Data

accu 2013

Lightning Talks

Pete Goodliffe – Design Sins

Alan Griffiths – The ACCU: Magic Happens Here

Dirk Haun – What's our Status?

Gail Ollis – Hello. I'm Back

Michel Grootjans – The Librarian

Björn Fahller – Why Are (only) We Here?

Frank Birbacher – Simple Quick Sort in C++

Calum Grant – C++ Active Objects

Didier Verna – Communities

Burkhard Kloss – Just a Minute

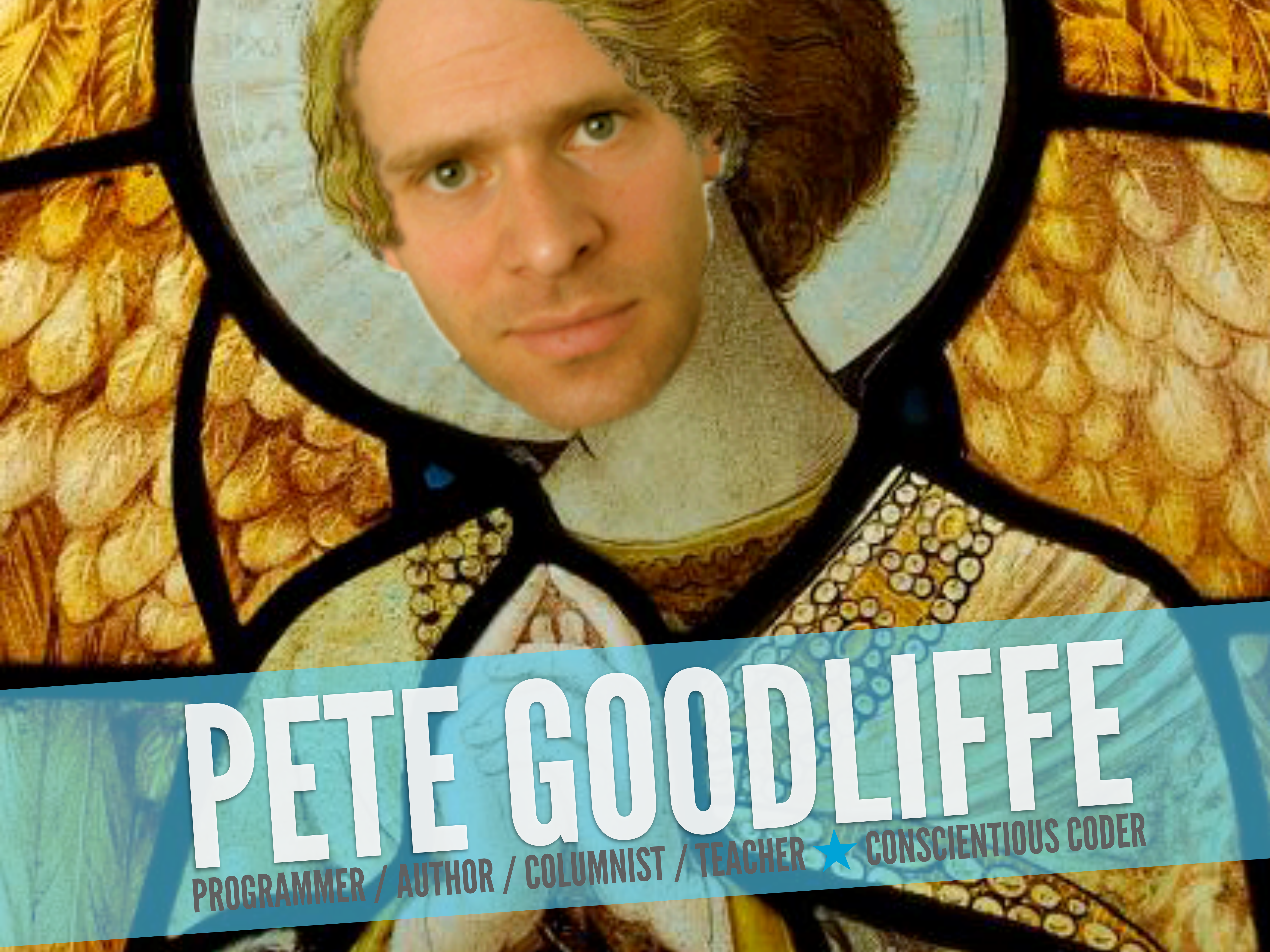
Guy Bolton King – BDD with Boost Test

Ed Sykes – A Decision Made With Data

accu 2013
bristol, april

designsins

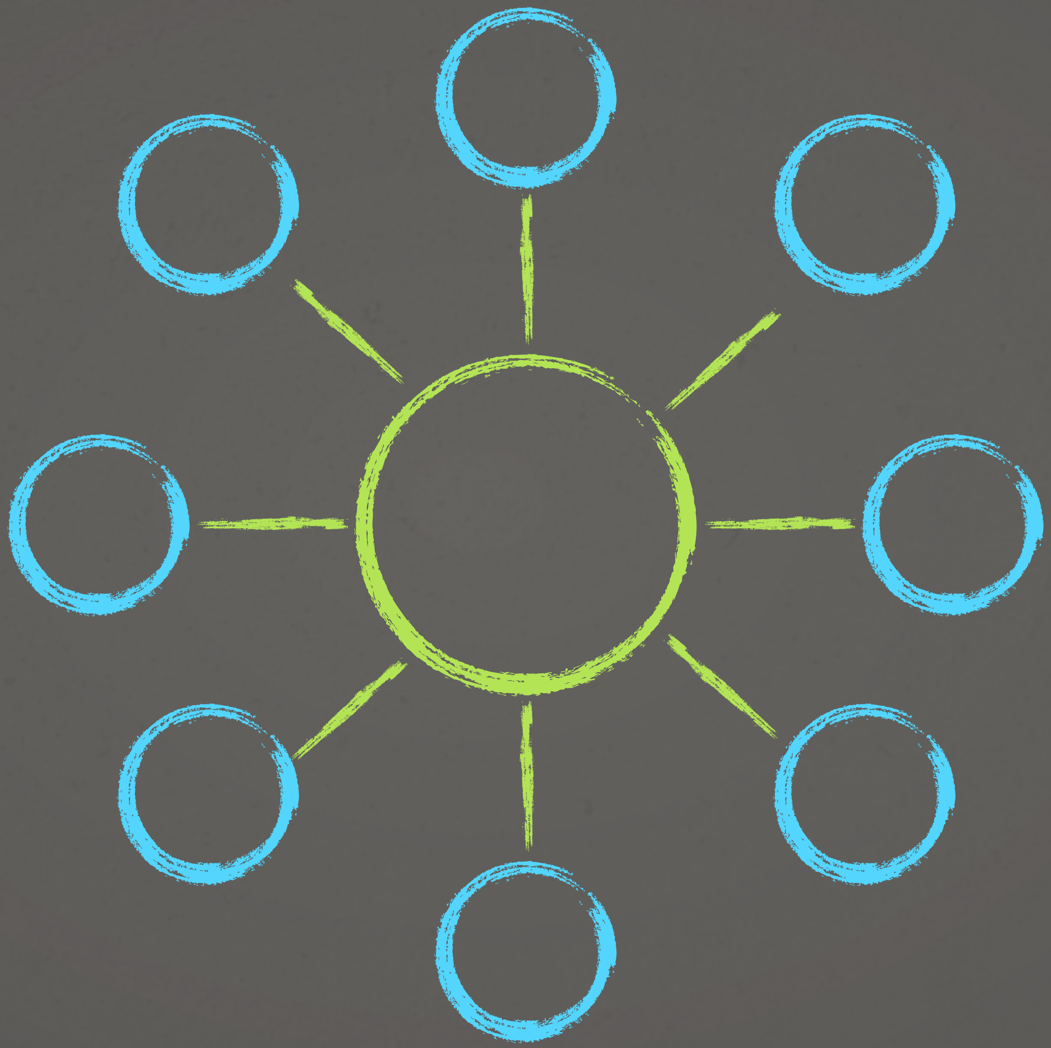
PETE GOODLIFFE pete@goodliffe.net @petegoodliffe

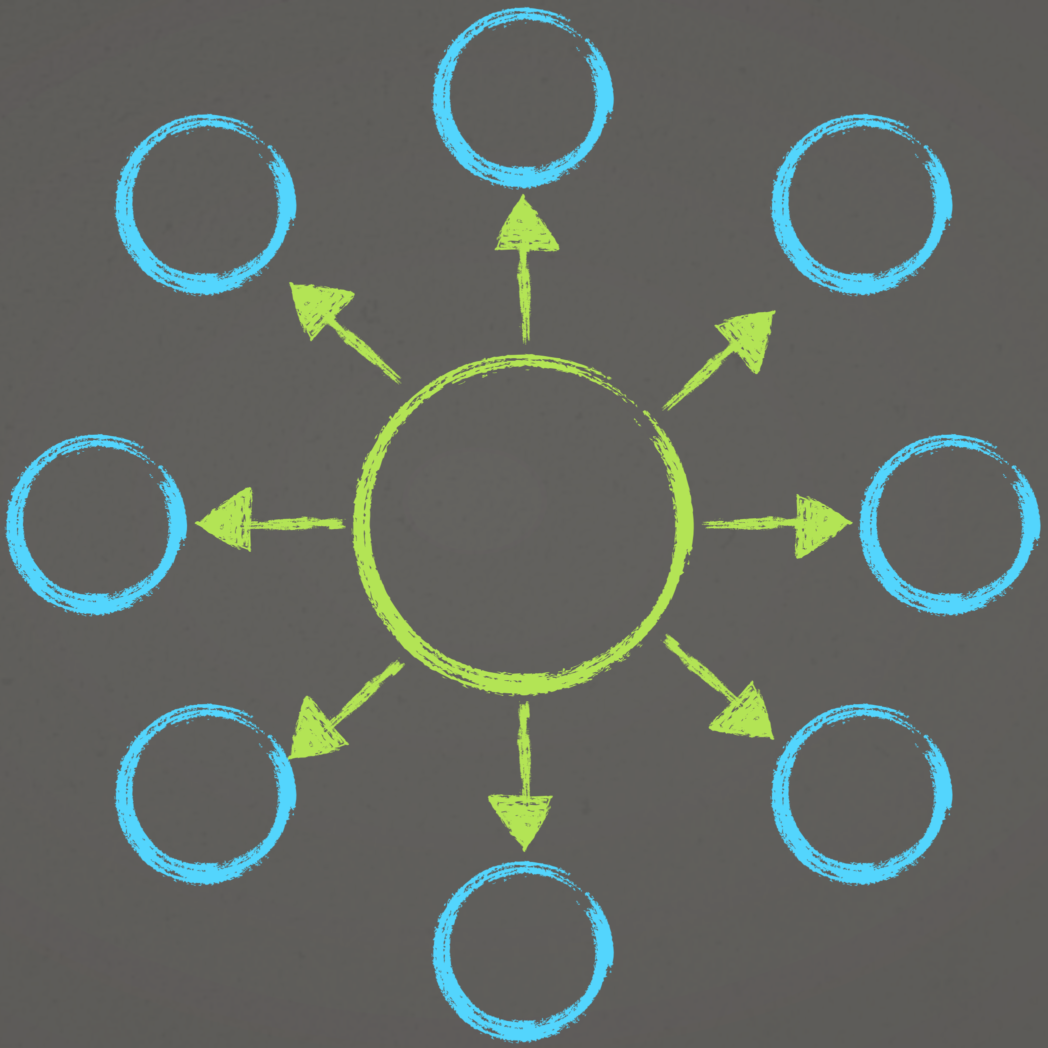


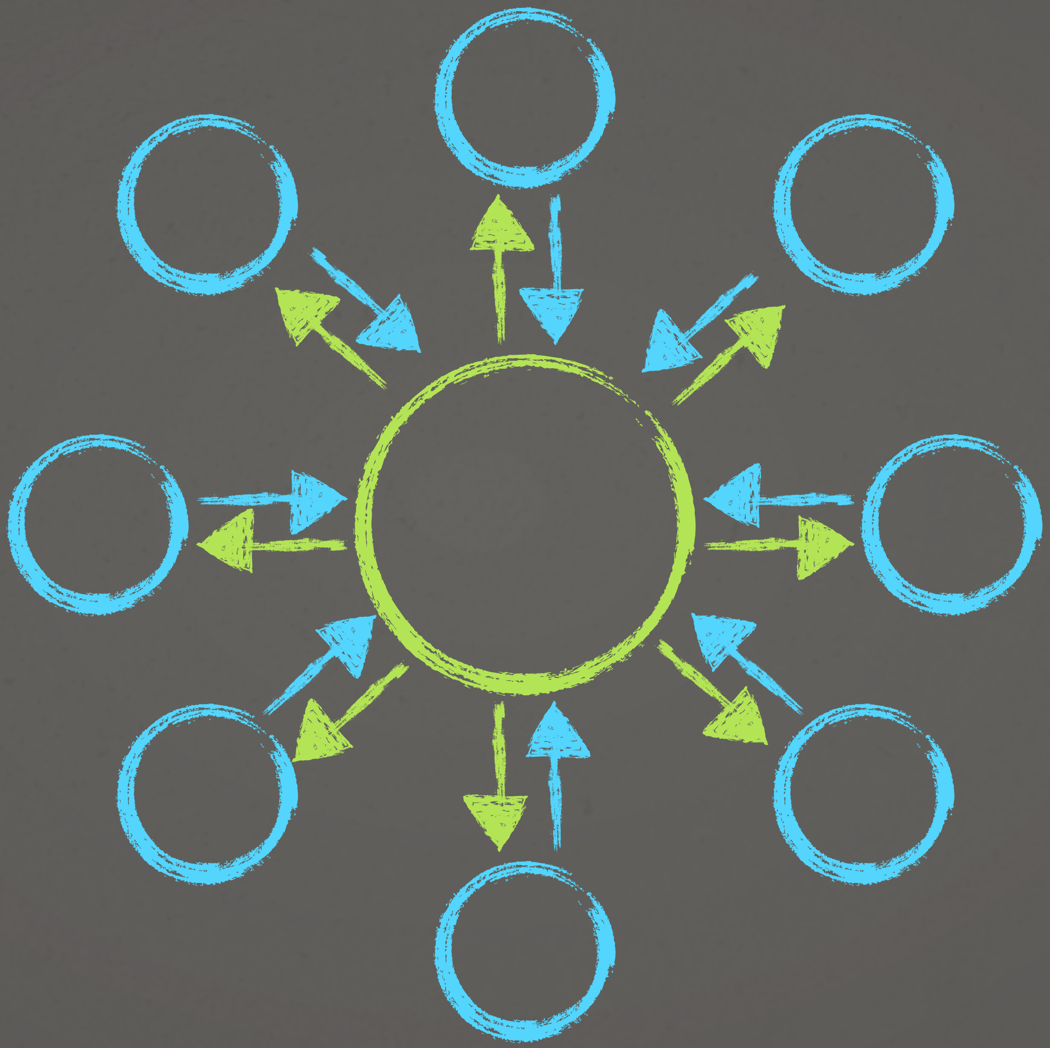
PETE GOODLIFFE

PROGRAMMER / AUTHOR / COLUMNIST / TEACHER ★ CONSCIENTIOUS CODER

exhibit **a**







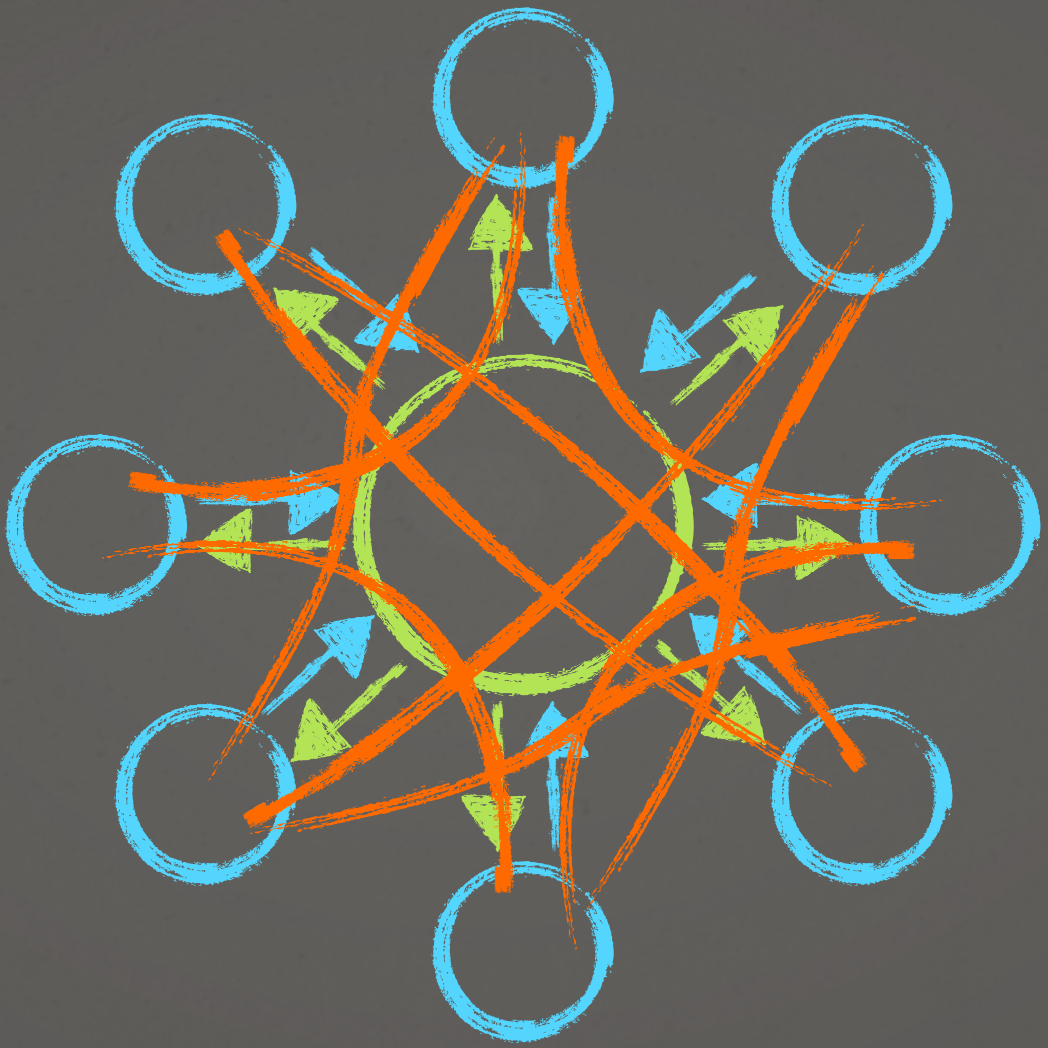
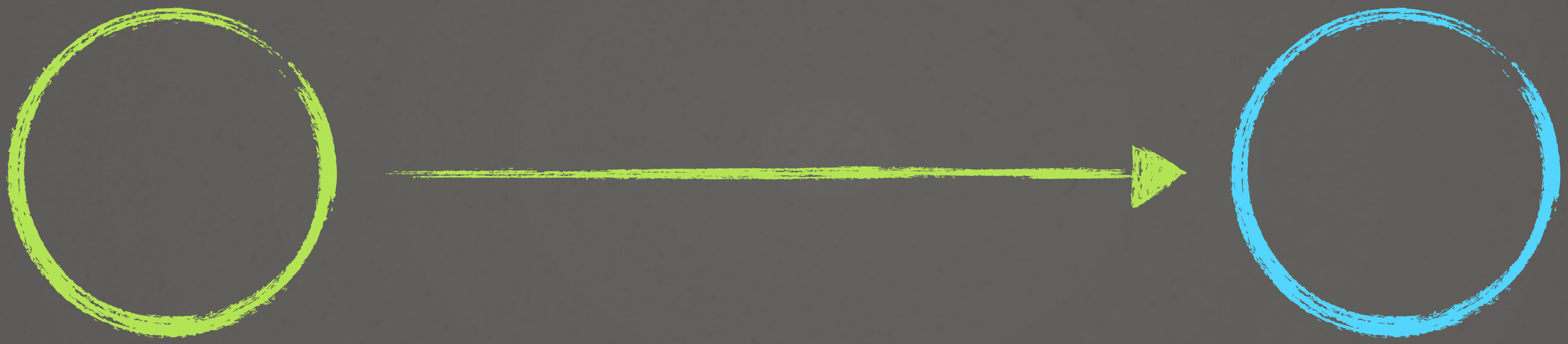
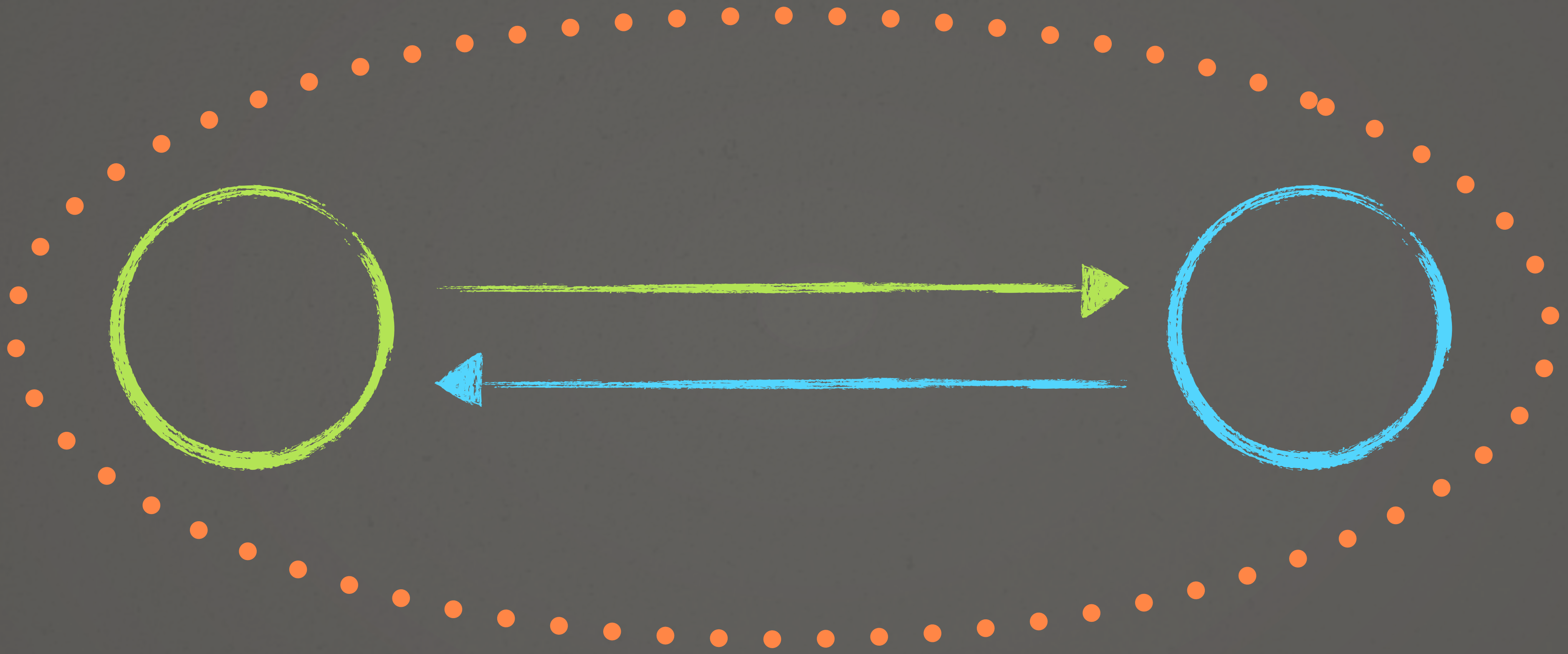


exhibit **b**





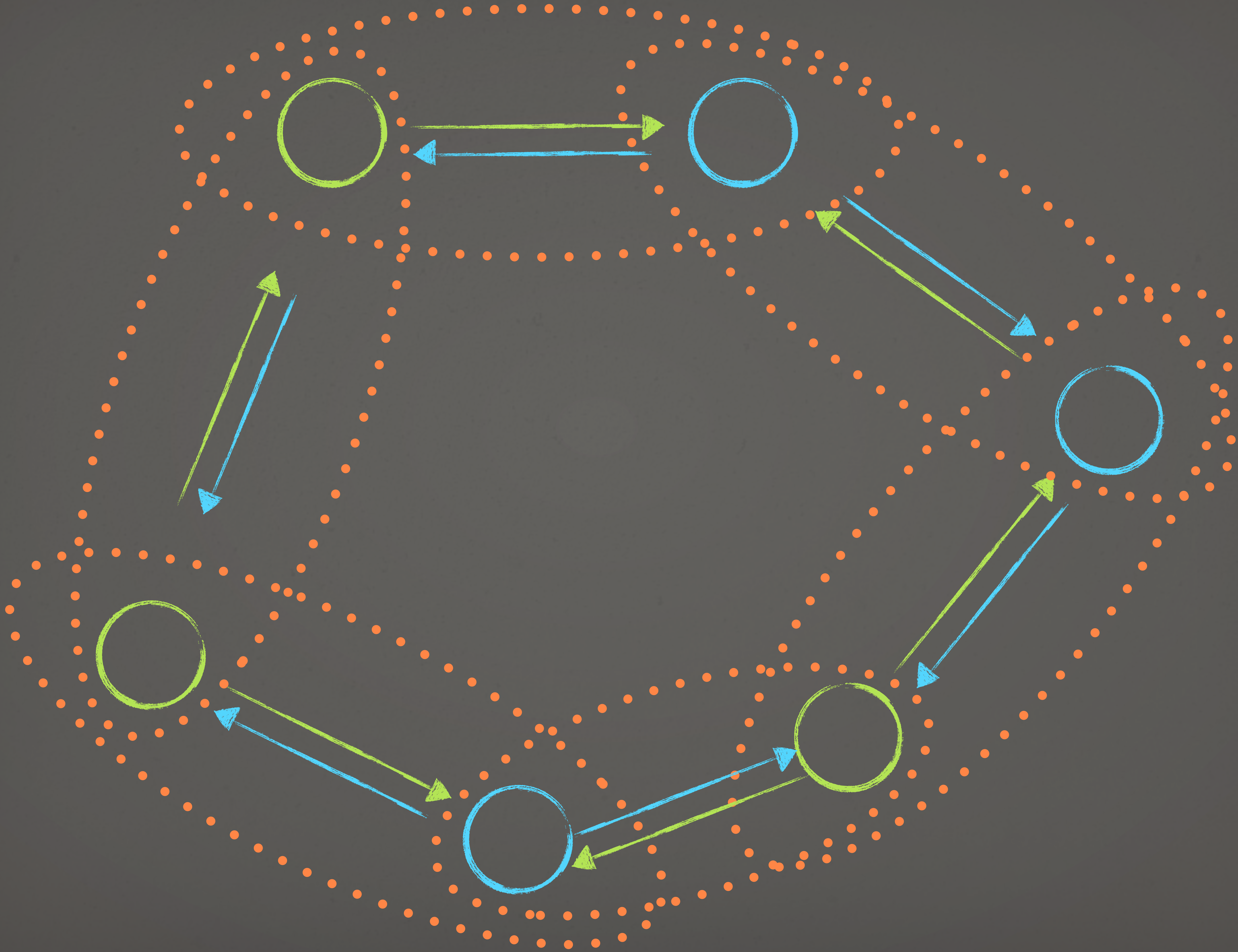
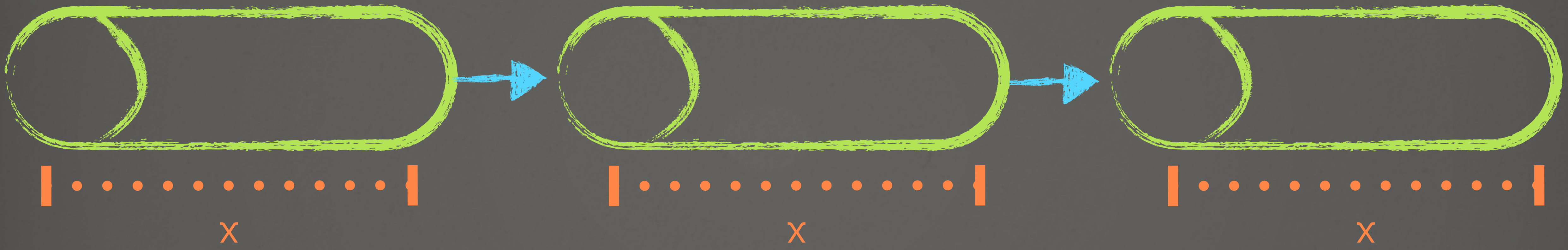
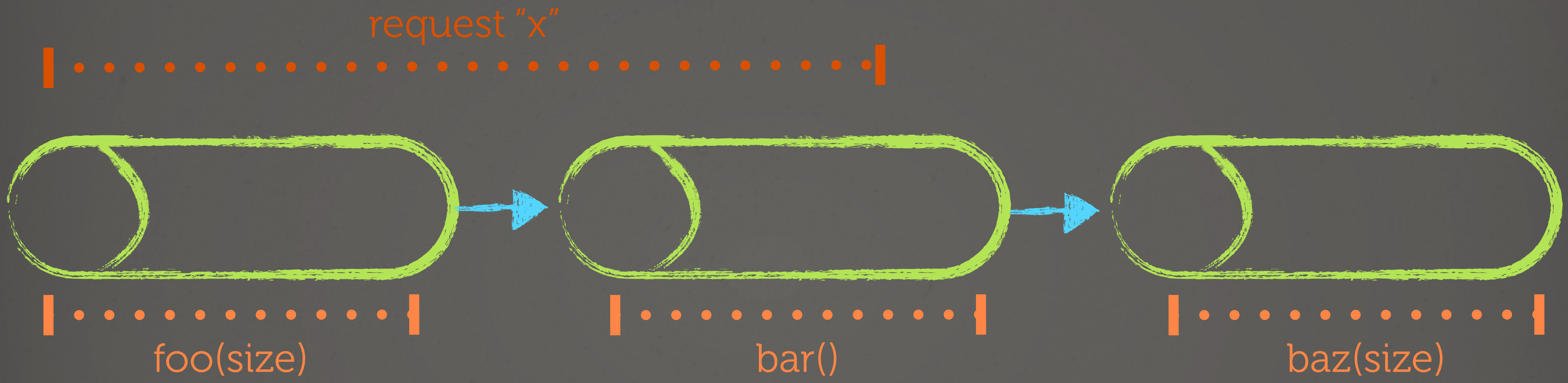


exhibit **c**


x
x
x


request "x"
foo(size)
bar()
baz(size)

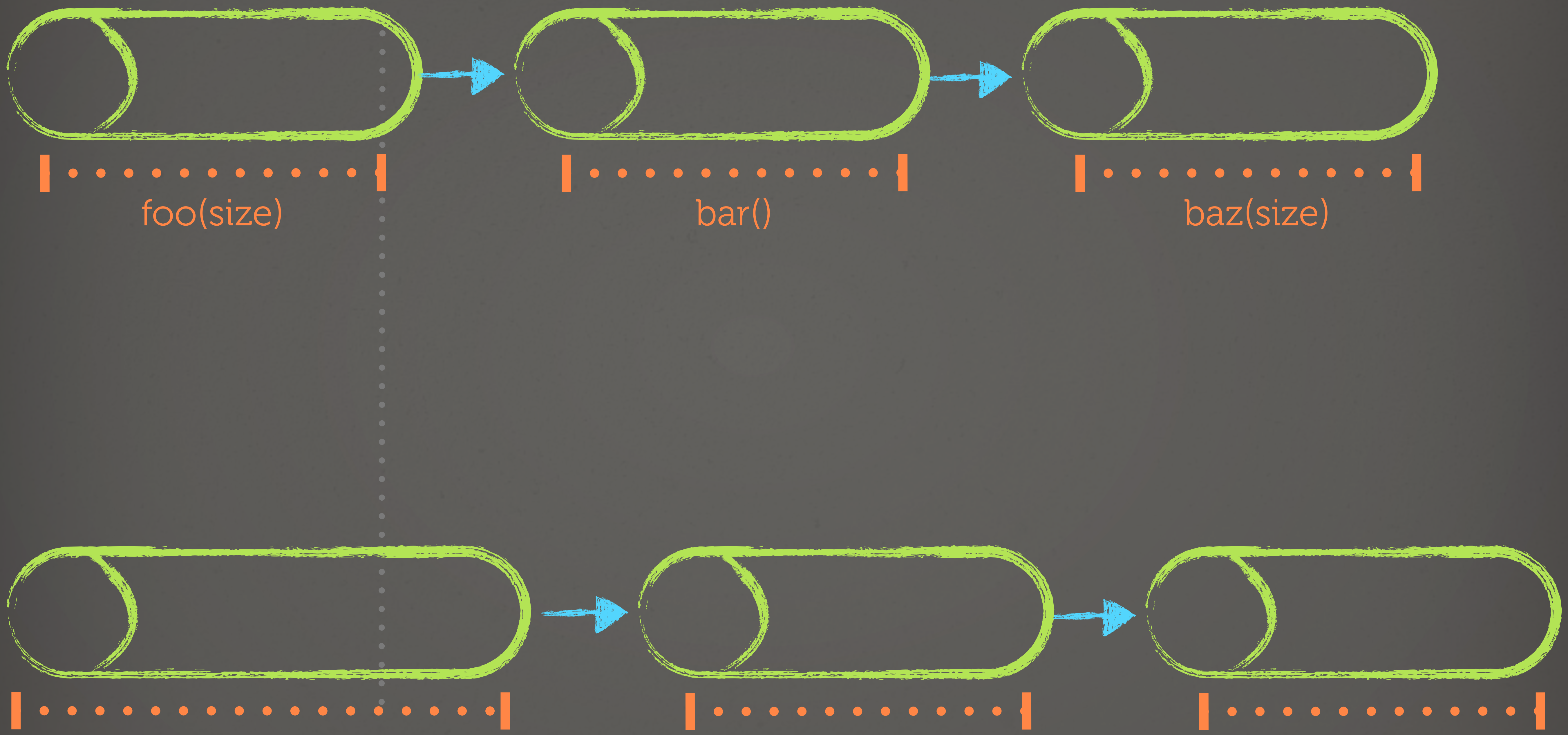

foo(size)
bar()
baz(size)

exhibit **d**

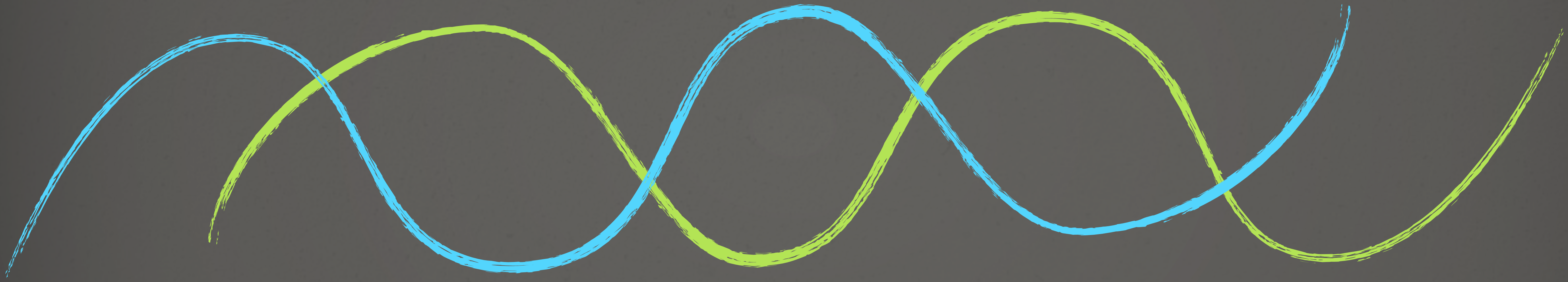
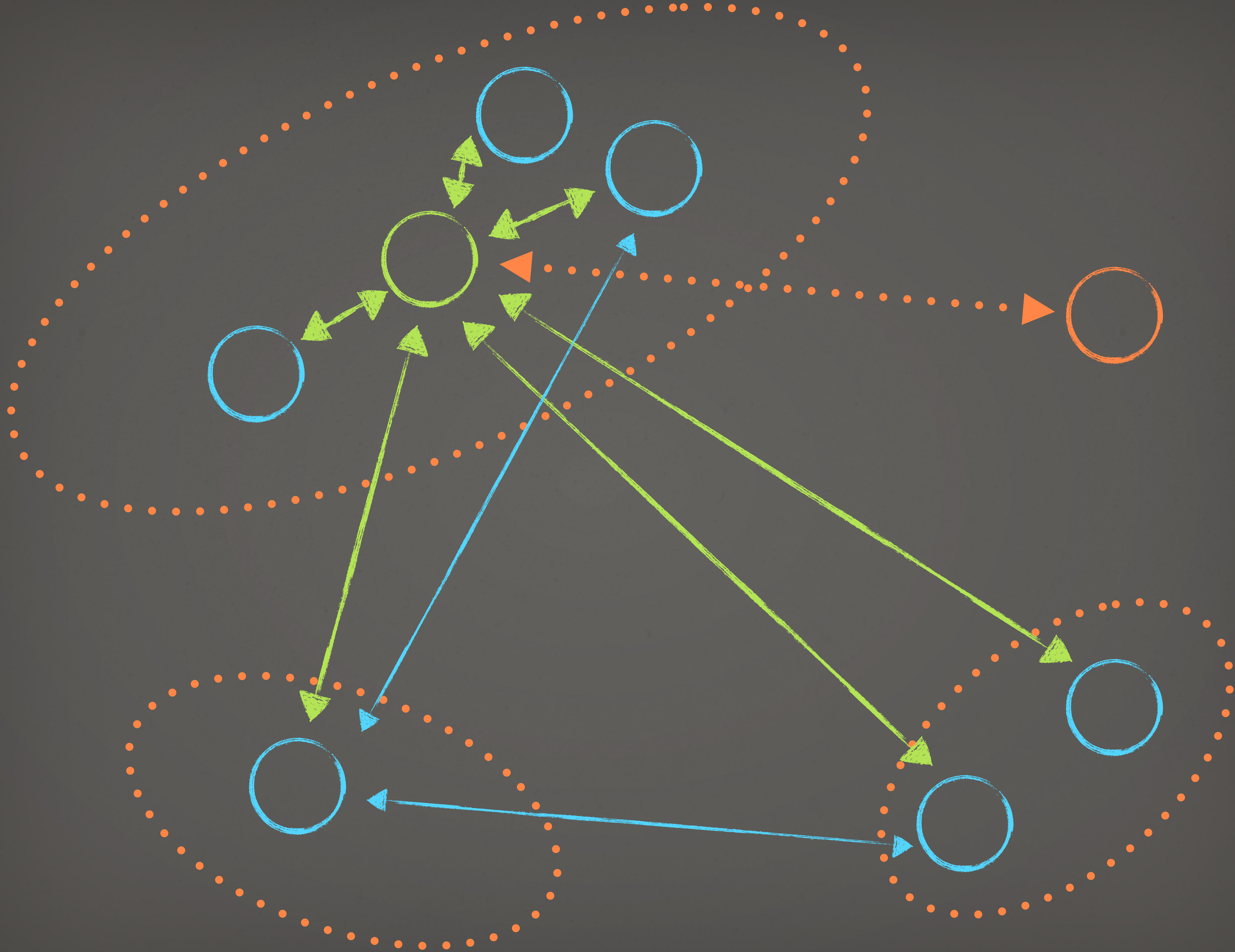


exhibit **e**

lack of generality
unnecessary generality
ambiguous ownership
subverting existing patterns

final **observation**



F I N

★ ★ ★

Pete Goodliffe @petegoodliffe pete@goodliffe.net

★★★

THEY ALL LIVED HAPPILY
E V E R A F T E R

★★★

(after mandatory ritual suicide)

IMAGE CREDITS

COD http://www.sxc.hu/photo/202410

Lightning Talks

Pete Goodliffe – Design Sins

Alan Griffiths – The ACCU: Magic Happens Here

Dirk Haun – What's our Status?

Gail Ollis – Hello. I'm Back

Michel Grootjans – The Librarian

Björn Fahller – Why Are (only) We Here?

Frank Birbacher – Simple Quick Sort in C++

Calum Grant – C++ Active Objects

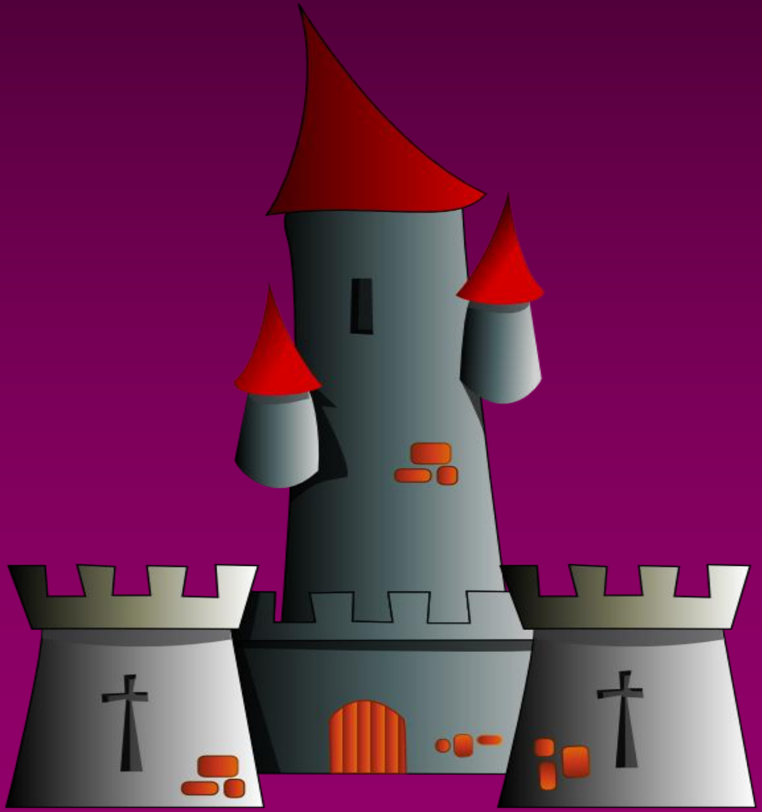
Didier Verna – Communities

Burkhard Kloss – Just a Minute

Guy Bolton King – BDD with Boost Test

Ed Sykes – A Decision Made With Data

The ACCU: Magic Happens Here



Alan Griffiths

alan@octopull.co.uk

@alanatoctopull

Who am I?

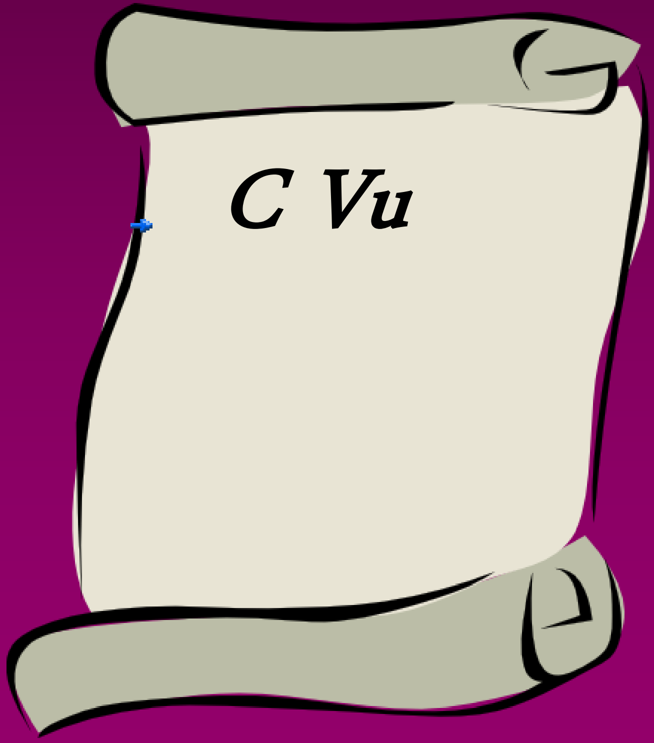
Alan Griffiths is a regular at the ACCU conference and has been developing software through many fashions in development processes, technologies, and programming languages.

During that time he's delivered working software and development processes, written contributions for magazines and books, spoken at a number of conferences and made many friends.

Firmly convinced that common sense is a rare and marketable commodity he's currently working as an independent through his company: Octopull Limited. (http://www.octopull.co.uk/)

The Beginning

- The C User Group (UK)
  - About C
  - C Vu newsletter
    - Irregular
    - Not much content
  - No website
  - No mailing lists
  - No conference

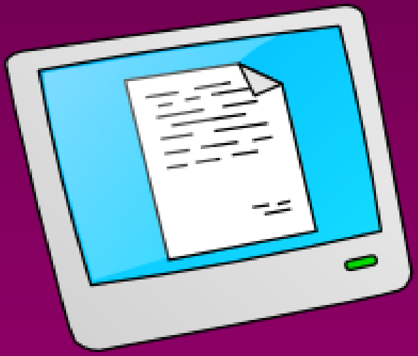


Add a little Francis

- C Vu every two months
- Incorporate the Borland C++ User Group
- Renamed “Association of C and C++ Users”
- Website
- Mailing lists
- Overload became C++ SIG journal
- ISDF Journal
- Early conferences

The 21st Century

- C Vu & Overload professionally printed
- Overload
  - Editorial team – improved quality
  - Broader, “professional” focus
- Conference
  - Professional organisers
- Mentored project lists
- Local group meetings
- Recent: C Vu follows Overload model



The Future

- ACCU is for:
  - Finding other people who will stimulate, enthuse or enable becoming a better programmer
  - Socialising with other geeks (preferably under the influence of alcohol)
  - Programming tips, techniques, craft and lore
  - Discussion of programming languages (except VB and, possibly, Perl but particularly C++)
- A bit like StackOverflow?

Don't Rely on "Magic Happens Here"

- If you look at any activity, process, or discipline from far enough away it looks simple.
  - That's true of programming and of the ACCU
- On any project there are likely many things that an individual doesn't get actively involved in
  - That's true of programming and of the ACCU
- You don't have to understand all the magic that makes it work, but it doesn't hurt to understand some of it.
  - That's true of programming and of the ACCU
- http://programmer.97things.oreilly.com/wiki/index.php/Don%27t_Rely_on_%22Magic_Happens_Here%22

accu 2013

Lightning Talks

Pete Goodliffe – Design Sins

Alan Griffiths – The ACCU: Magic Happens Here

Dirk Haun – What's our Status?

Gail Ollis – Hello. I'm Back

Michel Grootjans – The Librarian

Björn Fahller – Why Are (only) We Here?

Frank Birbacher – Simple Quick Sort in C++

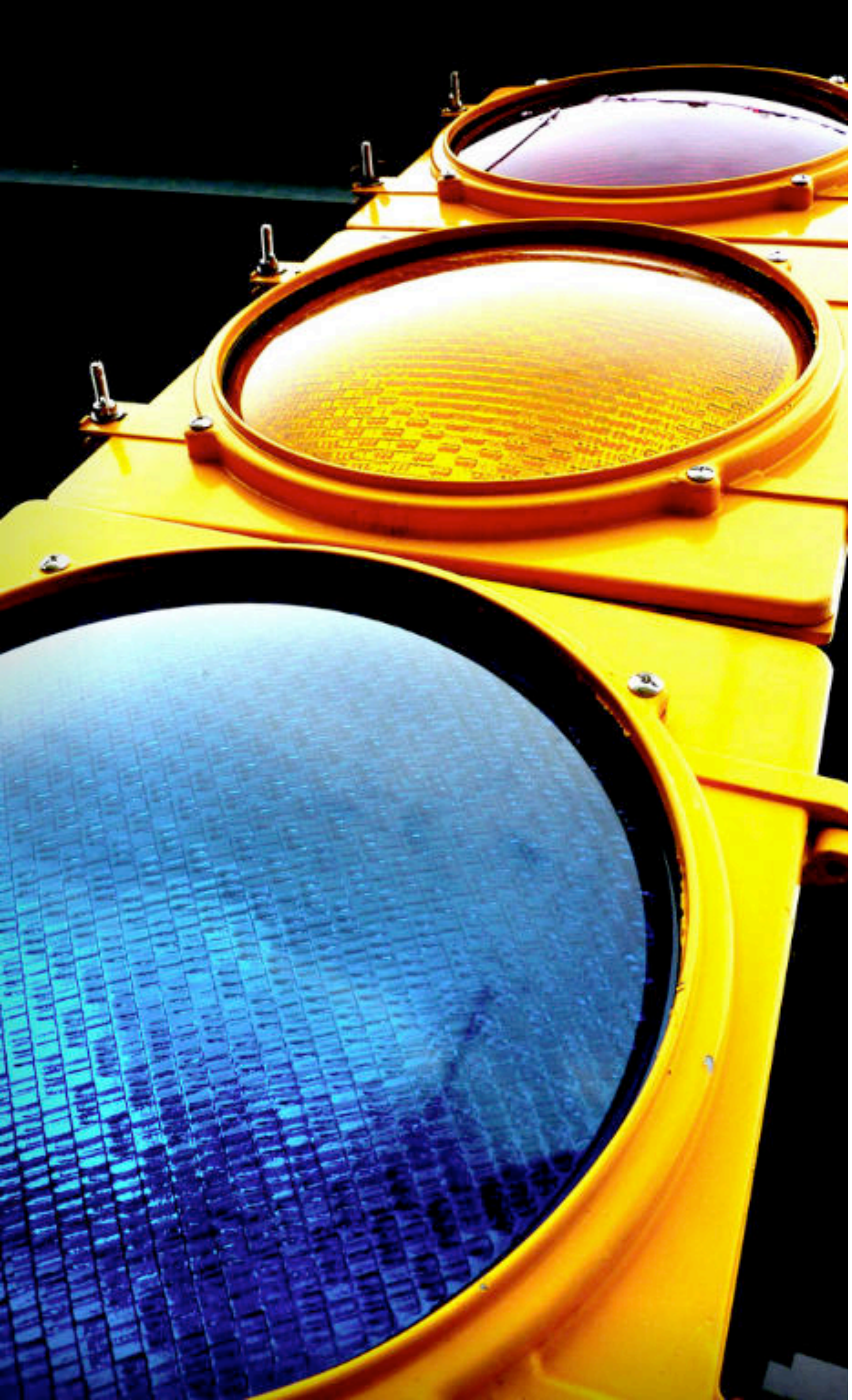
Calum Grant – C++ Active Objects

Didier Verna – Communities

Burkhard Kloss – Just a Minute

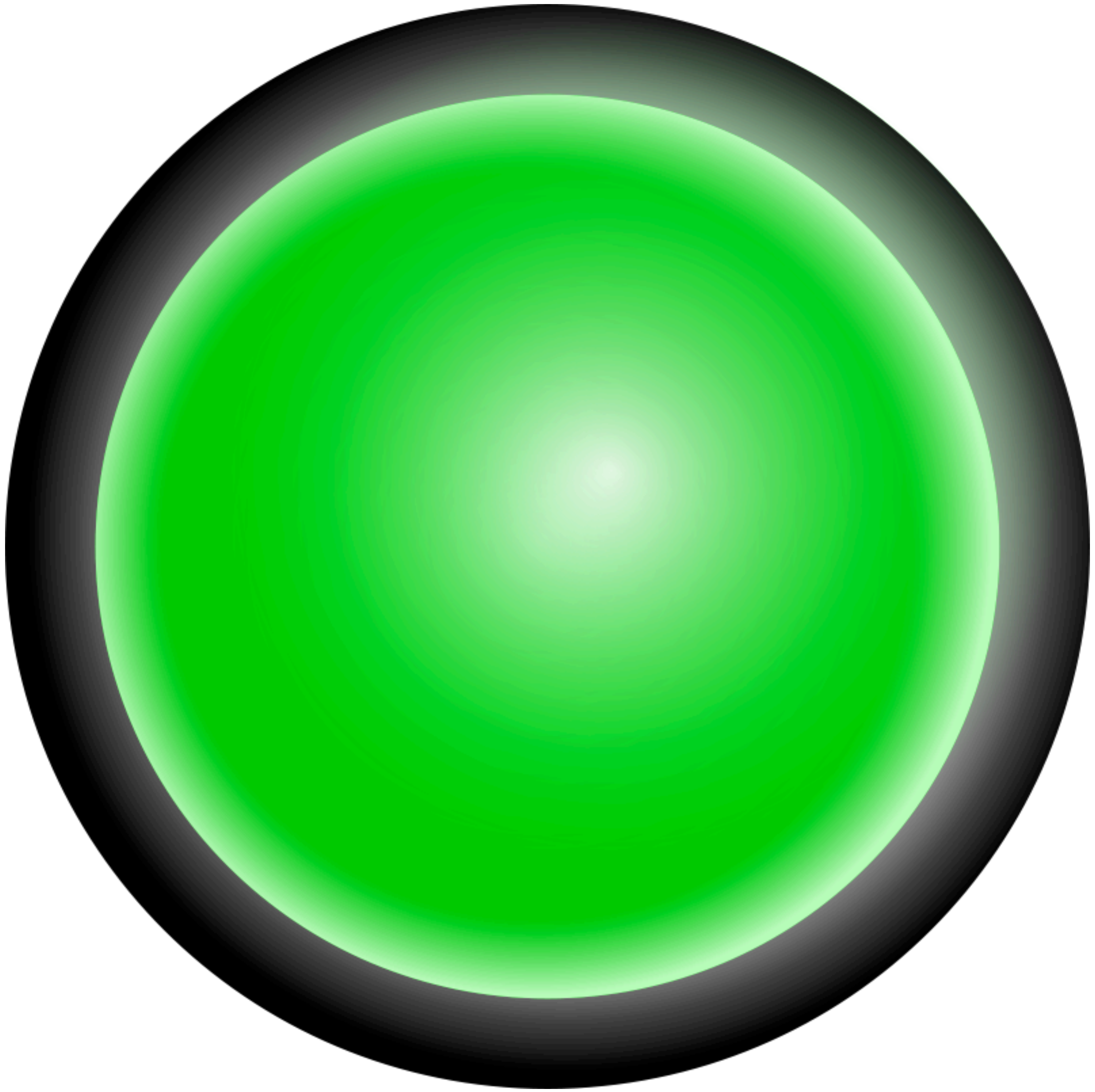
Guy Bolton King – BDD with Boost Test

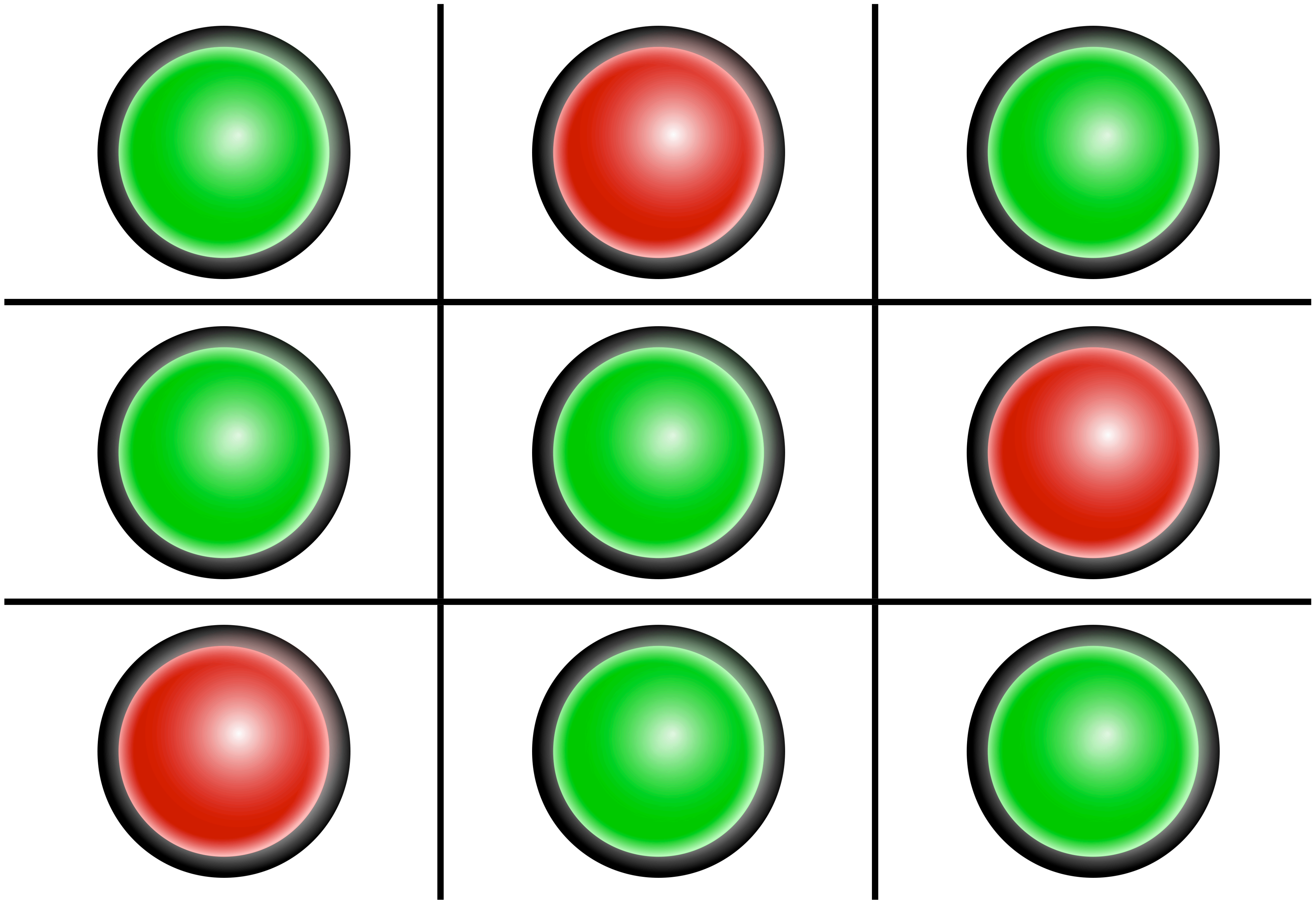
Ed Sykes – A Decision Made With Data

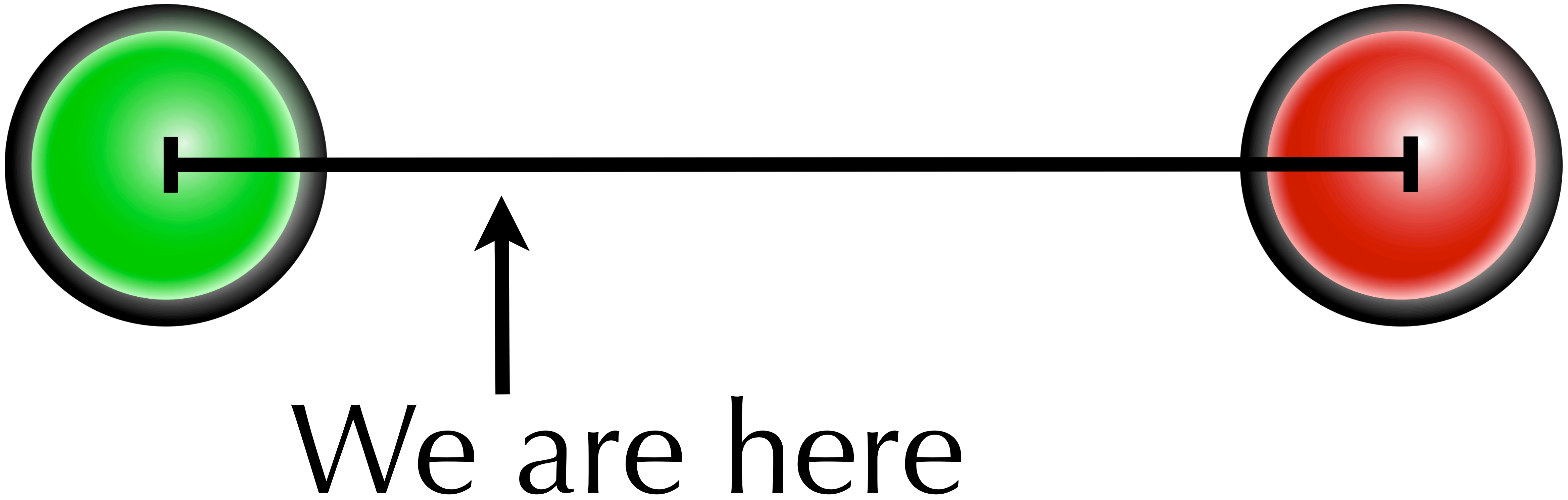


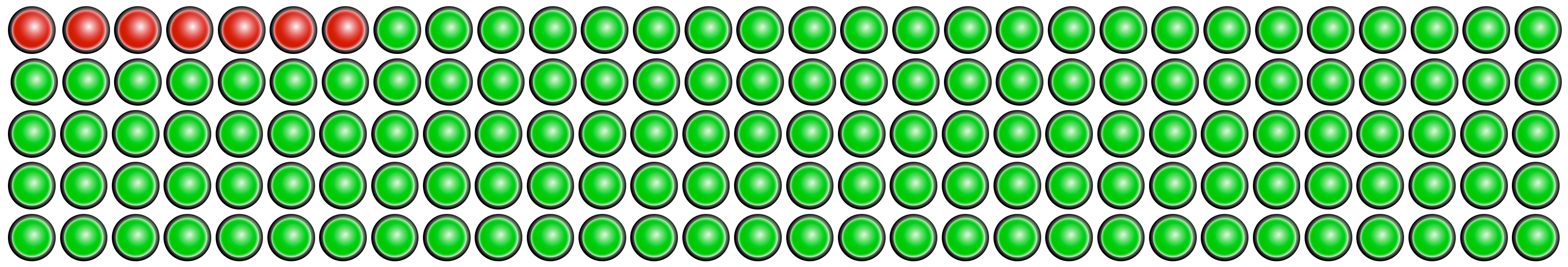
What's our Status?

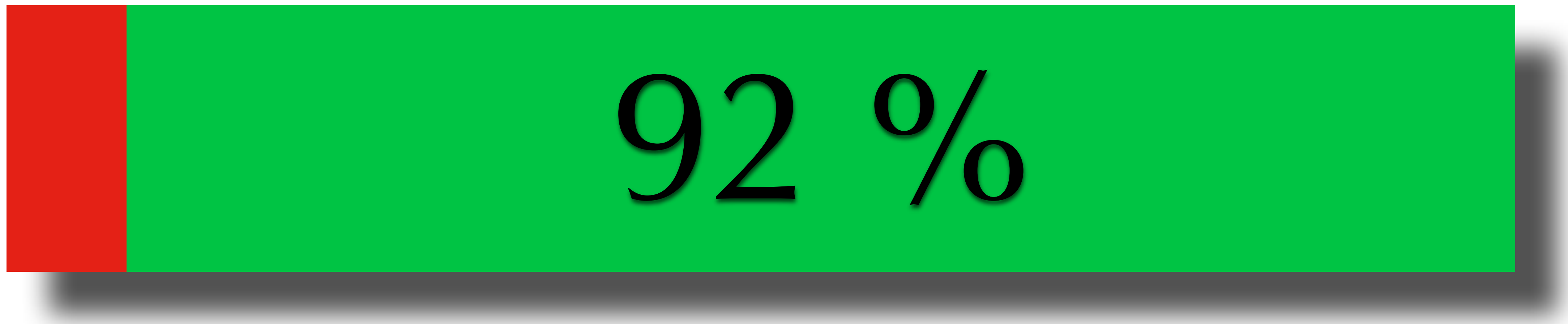
Dirk Haun,
ACCU 2013



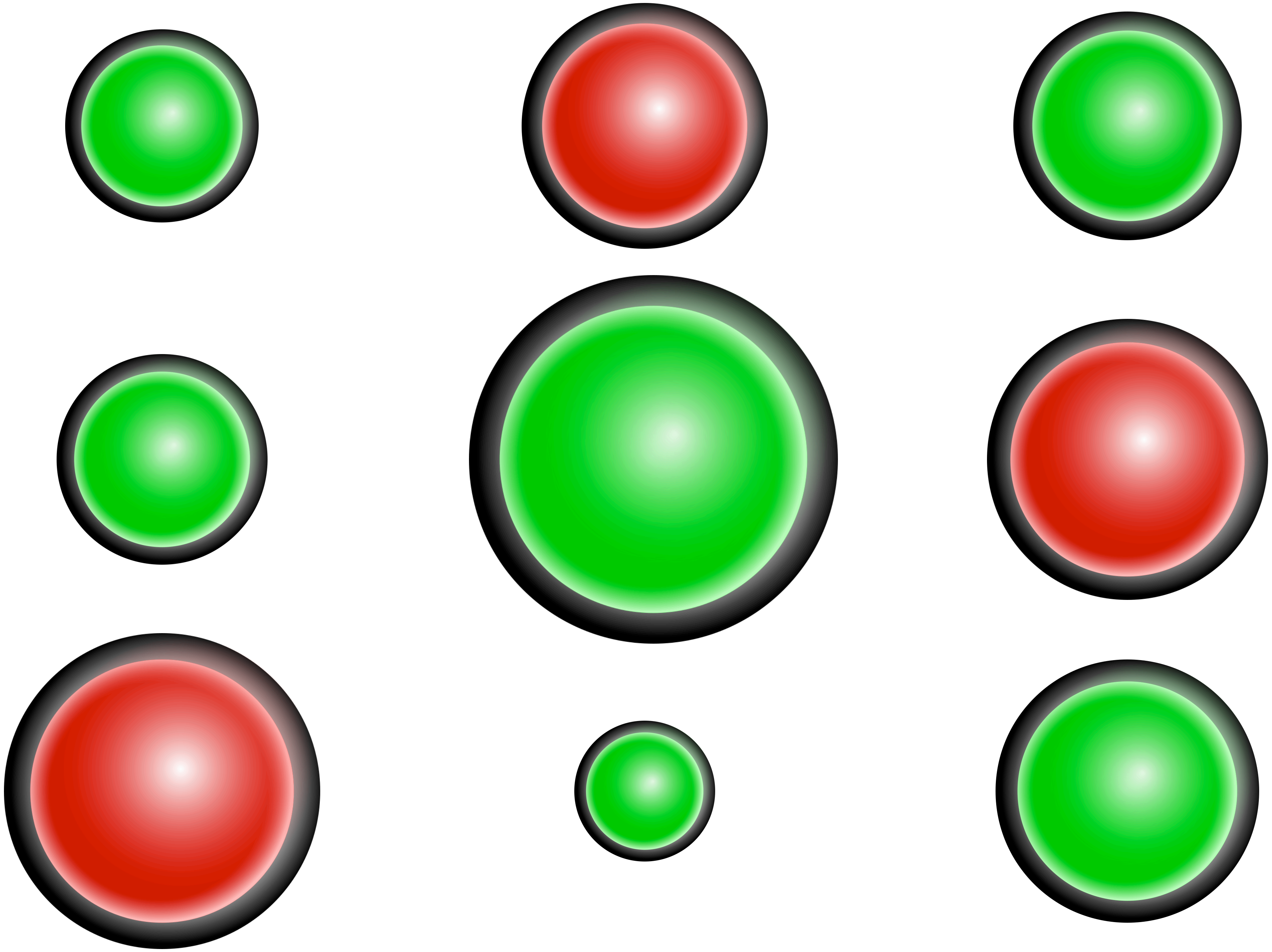



We are here




92 %

What about importance?



Boss

Support

Sysadmin

No one solution

Other industries?

I can has help plz?

dirk@haun-online.de
@dirkhaun
www.themobilepresenter.com

Credits

Photos and images, in order of appearance:
Lights! by Lawrence Rayner (Flickr),
LED, Green and LED, Red by Bryan Nielsen (Open Clipart Library),
Close up / Macro of four felt-tip-pencils in green, yellow, blue and orange by photosteve101 (Flickr),
[22.365] sphere-itize me, captain by db Photography | Demi-Brooke (Flickr),
Business Hands by Oleg Prikhodko (iStockphoto File #379024),
How may I help you? by Exact Image Limited (iStockphoto File #345416),
Network technician performs preventive maintenance a server by maximili (iStockphoto File #20452635),
Rings by Les Chatfield (Flickr),
Control Room by Alaa Hamed (stock.xchng),
(original source of the "help plz?" cat photo is unknown - please contact me if you have more information),
Hello World by Jessica Bergs & Dirk Haun.

accu 2013 Lightning Talks

Pete Goodliffe – Design Sins

Alan Griffiths – The ACCU: Magic Happens Here

Dirk Haun – What's our Status?

Gail Ollis – Hello. I'm Back

Michel Grootjans – The Librarian

Björn Fahller – Why Are (only) We Here?

Frank Birbacher – Simple Quick Sort in C++

Calum Grant – C++ Active Objects

Didier Verna – Communities

Burkhard Kloss – Just a Minute

Guy Bolton King – BDD with Boost Test

Ed Sykes – A Decision Made With Data

accu 2013

Lightning Talks

Pete Goodliffe – Design Sins

Alan Griffiths – The ACCU: Magic Happens Here

Dirk Haun – What's our Status?

Gail Ollis – Hello. I'm Back

Michel Grootjans – The Librarian

Björn Fahller – Why Are (only) We Here?

Frank Birbacher – Simple Quick Sort in C++

Calum Grant – C++ Active Objects

Didier Verna – Communities

Burkhard Kloss – Just a Minute

Guy Bolton King – BDD with Boost Test

Ed Sykes – A Decision Made With Data

The librarian

ACCU 2013 Lightning Talk

#michelgrootjans

Ook

A programming language designed for orang-utans

Ook: design principles

Ook: design principles

A programming language should be writable and readable by orang-utans.

Ook: design principles

A programming language should be writable and readable by orang-utans.

To this end, the syntax should be simple, easy to remember, and not mention the word "monkey”.

Ook: design principles

A programming language should be writable and readable by orang-utans.

To this end, the syntax should be simple, easy to remember, and not mention the word "monkey".

Bananas are good.

Ook: Syntax elements

Ook: Syntax elements

Ook.

Ook: Syntax elements

Ook.

Ook!

Ook: Syntax elements

Ook.

Ook!

Ook?

Ook: Commands

Command	Symbol	Meaning
Ook. Ook?	>	increment the pointer
Ook? Ook.	<	decrement the pointer
Ook. Ook.	+	increment the integer at the pointer
Ook! Ook!	-	decrement the integer at the pointer
Ook! Ook.	.	output the ASCII character from the integer at the pointer
Ook. Ook!	,	input to the integer at the pointer (ASCII).
Ook! Ook?	[	jump forward to the statement after the corresponding Ook? Ook! if the byte at the pointer is zero
Ook? Ook!	]	jump back to the statement after the corresponding Ook! Ook? if the byte at the pointer is nonzero

Ook: Comments

Since the word "ook" can convey entire ideas, emotions, and abstract thoughts depending on the nuances of inflection, Ook! has no need of comments. The code itself serves perfectly well to describe in detail what it does and how it does it.

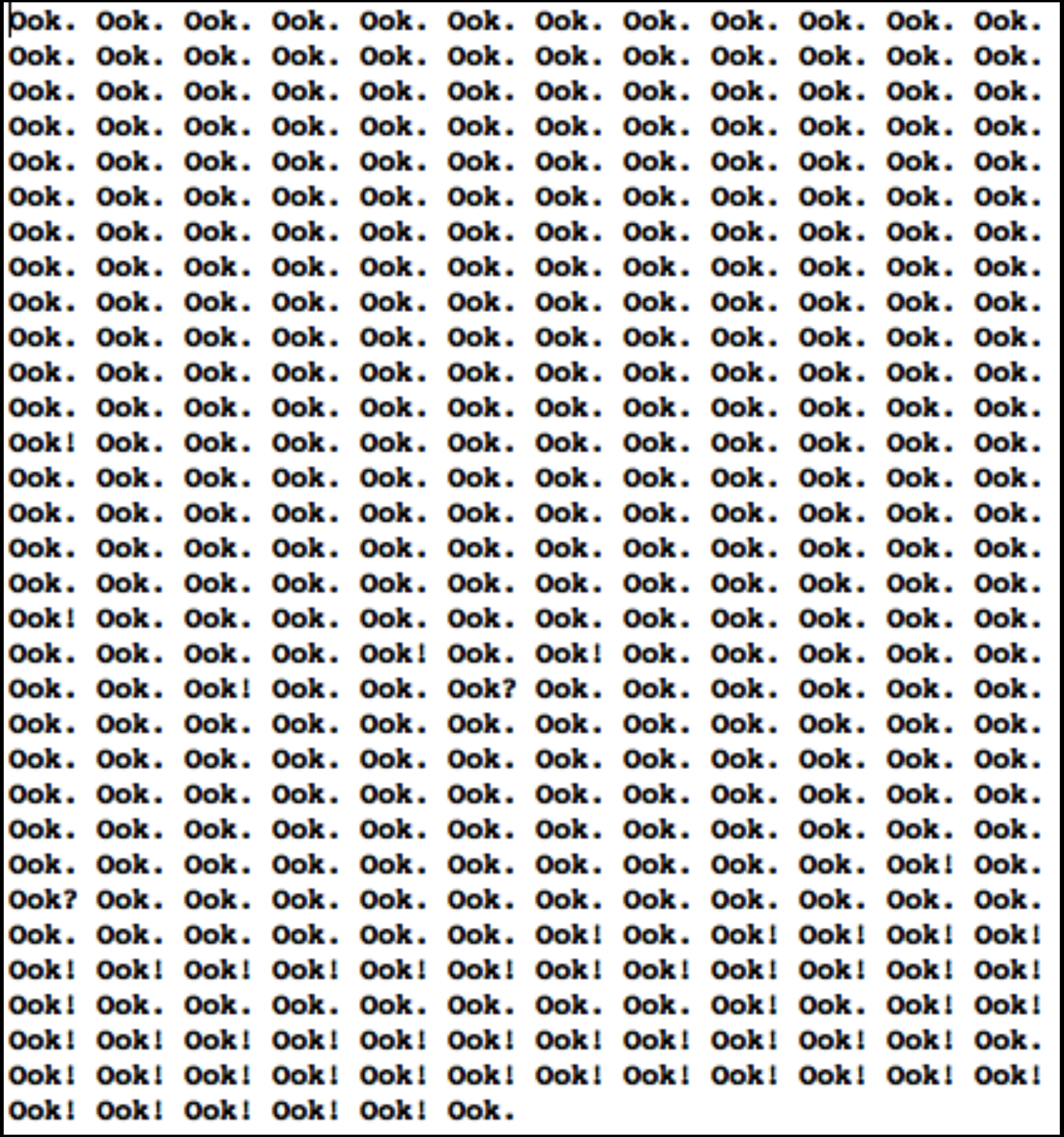
Ook: Comments

Since the word "ook" can convey entire ideas, emotions, and abstract thoughts depending on the nuances of inflection, Ook! has no need of comments. The code itself serves perfectly well to describe in detail what it does and how it does it.

Provided you are an orang-utan.

Let’s see some code

Guess what this code does



```
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook! Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook! Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook! Ook. Ook! Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook! Ook. Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook.
Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook. Ook! Ook! Ook! Ook!
Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook!
Ook! Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook. Ook! Ook!
Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook.
Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook!
Ook! Ook! Ook! Ook! Ook! Ook.
```

“Any monkey can write code that a computer can understand.

Good apes write code that an orang-utang can understand.”

-- The Librarian

Shameless plug

Join me tomorrow to learn a little about Ruby and Rails

#michelgrootjans

Lightning Talks

Pete Goodliffe – Design Sins
Alan Griffiths – The ACCU: Magic Happens Here
Dirk Haun – What's our Status?
Gail Ollis – Hello. I'm Back
Michel Grootjans – The Librarian
Björn Fahller – Why Are (only) We Here?
Frank Birbacher – Simple Quick Sort in C++
Calum Grant – C++ Active Objects
Didier Verna – Communities
Burkhard Kloss – Just a Minute
Guy Bolton King – BDD with Boost Test
Ed Sykes – A Decision Made With Data

accu 2013

Lightning Talks

Pete Goodliffe – Design Sins

Alan Griffiths – The ACCU: Magic Happens Here

Dirk Haun – What's our Status?

Gail Ollis – Hello. I'm Back

Michel Grootjans – The Librarian

Björn Fahller – Why Are (only) We Here?

Frank Birbacher – Simple Quick Sort in C++

Calum Grant – C++ Active Objects

Didier Verna – Communities

Burkhard Kloss – Just a Minute

Guy Bolton King – BDD with Boost Test

Ed Sykes – A Decision Made With Data

Simple Quick Sort in C++

Frank Birbacher

April 10, 2013

QuickSort

3 1 4 2

QuickSort

1 2 3 4

Steps to take

- select pivot
- partition
- recurse

Signature

```
template<typename Iter>
void quick_sort(
                Iter first,
                Iter last
        )
{
```

Select Pivot I

Iter **const** pivot = first++;

Select Pivot II

```
if(first == last) return;
Iter const pivot = first++;
if(first == last) return;
```

Partition Predicate I

```
typedef typename std::iterator_traits<Iter>
        ::reference reference;

auto const lessThanPivot
        = [=](reference current)
        { return current < *pivot; };
```

Partition Predicate II

Boost:

```
using boost::lambda::_1;

auto const lessThanPivot
        = _1 < *pivot;
```

Partition

```
Iter const middle =
        std::partition(first, last, lessThanPivot);
```

Recurse

```
quick_sort(first, middle);
quick_sort(middle, last);
```

Place Pivot

```
std::rotate(pivot, first, middle);
```

The Whole Picture

```
template<typename Iter>
void quick_sort(Iter first, Iter last) {
  if(first == last) return;
  Iter const pivot = first++;
  if(first == last) return;

  auto const lessThanPivot = _1 < *pivot;
  Iter const middle = std::partition(first, last, lessThanPivo

  quick_sort(first, middle);
  quick_sort(middle, last);

  std::rotate(pivot, first, middle);
}
```

Lightning Talks

Pete Goodliffe – Design Sins

Alan Griffiths – The ACCU: Magic Happens Here

Dirk Haun – What's our Status?

Gail Ollis – Hello. I'm Back

Michel Grootjans – The Librarian

Björn Fahller – Why Are (only) We Here?

Frank Birbacher – Simple Quick Sort in C++

Calum Grant – C++ Active Objects

Didier Verna – Communities

Burkhard Kloss – Just a Minute

Guy Bolton King – BDD with Boost Test

Ed Sykes – A Decision Made With Data

accu 2013

Lightning Talks

Pete Goodliffe – Design Sins

Alan Griffiths – The ACCU: Magic Happens Here

Dirk Haun – What's our Status?

Gail Ollis – Hello. I'm Back

Michel Grootjans – The Librarian

Björn Fahller – Why Are (only) We Here?

Frank Birbacher – Simple Quick Sort in C++

Calum Grant – C++ Active Objects

Didier Verna – Communities

Burkhard Kloss – Just a Minute

Guy Bolton King – BDD with Boost Test

Ed Sykes – A Decision Made With Data

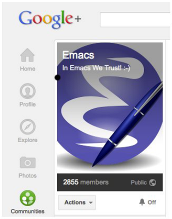
Communities

Didier Verna

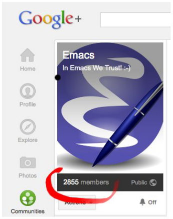
didier@didierverna.net
`facebook/didierverna`
`@didierverna`

ACCU 2013 – Wednesday, April 10th

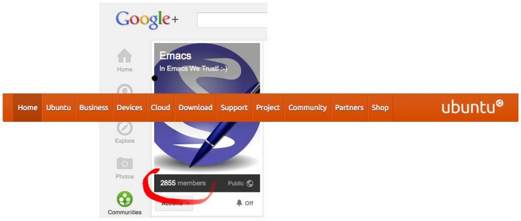
Communities



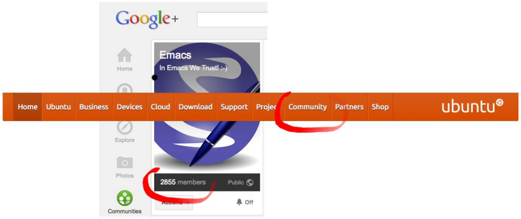
Communities



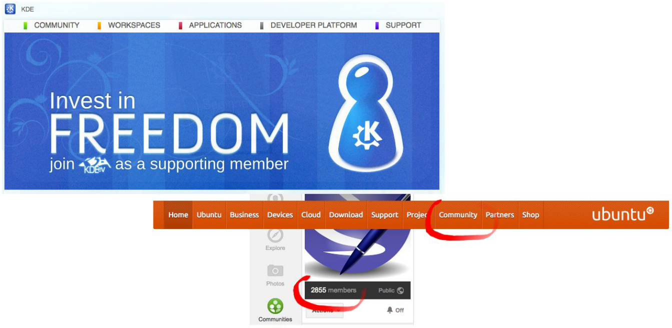
Communities



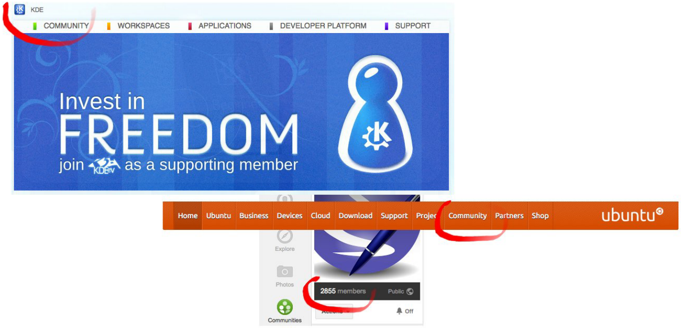
Communities



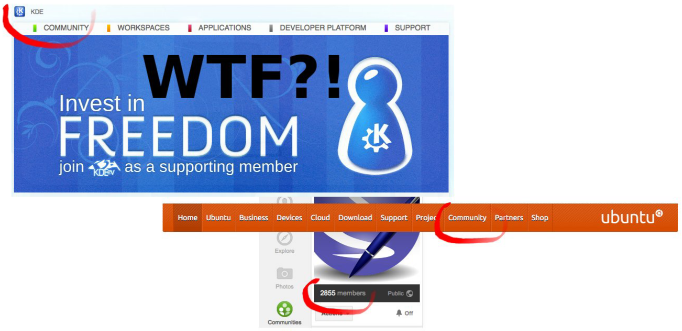
Communities



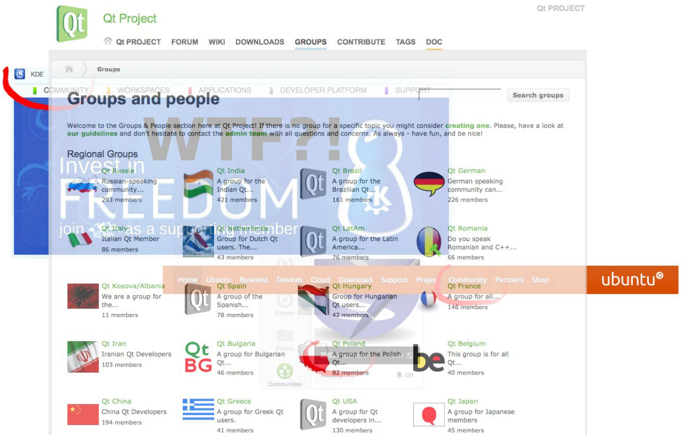
Communities



Communities



Communities



Qt Project

Qt PROJECT FORUM WIKI DOWNLOADS GROUPS CONTRIBUTE TAGS DOC

Qt PROJECT

Groups

Search groups

Groups and people

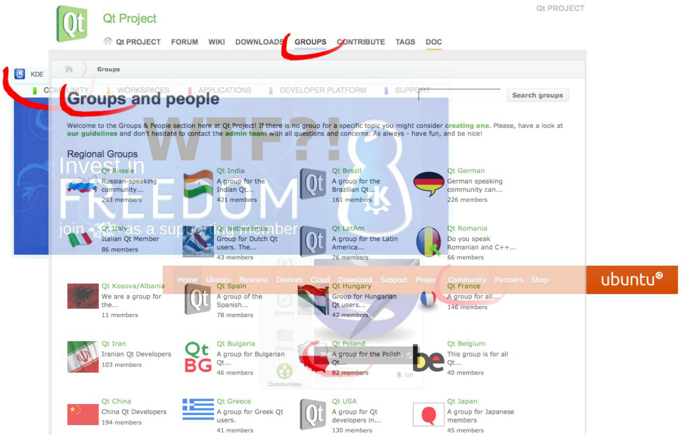
Welcome to the Groups & People section here at Qt Project! If there is no group for a specific topic you might consider **creating one**. Please, have a look at **our guidelines** and don't hesitate to contact the **admin team** with all questions and concerns. As always - have fun, and be nice!

Regional Groups

Qt Russia Russian-speaking community… 293 members	Qt India A group for the Indian Qt… 421 members	Qt Brazil A group for the Brazilian Qt… 161 members	Qt German German speaking community can… 226 members
Qt Italy Italian Qt Member 86 members	Qt Netherlands Group for Dutch Qt users. The… 43 members	Qt LatAm A group for the Latin America… 76 members	Qt Romania Do you speak Romanian and C++… 66 members
Qt Kosova/Albania We are a group for the… 11 members	Qt Spain A group of the Spanish… 78 members	Qt Hungary Group for Hungarian Qt users… 42 members	Qt France A group for all… 146 members
Qt Iran Iranian Qt Developers 103 members	Qt Bulgaria A group for Bulgarian Qt… 46 members	Qt Poland A group for the Polish Qt… 82 members	Qt Belgium This group is for all Qt… 40 members
Qt China China Qt Developers 194 members	Qt Greece A group for Greek Qt users. 41 members	Qt USA A group for Qt developers in… 130 members	Qt Japan A group for Japanese members 45 members

KDE
COMMUNITY
WORKSPACES
APPLICATIONS
DEVELOPER PLATFORM
SUPPORT
Invest in
FREEDOM
join us as a supporting member
WTF?!
Home Ubuntu Business Devices Cloud Download Support Project Community Partners Shop
ubuntu
Explore
Photos
Communities
Off

Communities


Qt PROJECT
Qt Project
Qt PROJECT FORUM WIKI DOWNLOADS GROUPS CONTRIBUTE TAGS DOC
Groups
KDE
COMMUNITY WORKSPACES APPLICATIONS DEVELOPER PLATFORM SUPPORT
Search groups
WTF?!
Invest in FREEDOM join KDE as a supporting member
Home Ubuntu Business Devices Cloud Download Support Project Community Partners Shop
ubuntu
Explore
Photos
Communities
Off

Groups and people

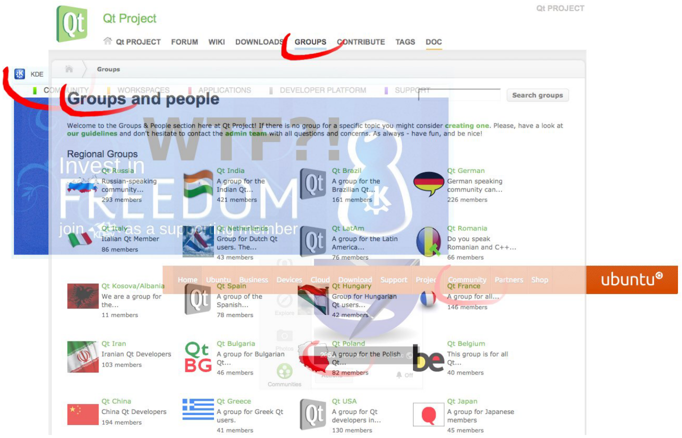
Welcome to the Groups & People section here at Qt Project! If there is no group for a specific topic you might consider **creating one**. Please, have a look at **our guidelines** and don't hesitate to contact the **admin team** with all questions and concerns. As always - have fun, and be nice!

Regional Groups

Qt Russia Russian-speaking community… 293 members	Qt India A group for the Indian Qt… 421 members	Qt Brazil A group for the Brazilian Qt… 161 members	Qt German German speaking community can… 226 members
Qt Italy Italian Qt Member 86 members	Qt Netherlands Group for Dutch Qt users. The… 43 members	Qt LatAm A group for the Latin America… 76 members	Qt Romania Do you speak Romanian and C++… 66 members
Qt Kosova/Albania We are a group for the… 11 members	Qt Spain A group of the Spanish… 78 members	Qt Hungary Group for Hungarian Qt users… 42 members	Qt France A group for all… 146 members
Qt Iran Iranian Qt Developers 103 members	Qt Bulgaria A group for Bulgarian Qt… 46 members	Qt Poland A group for the Polish Qt… 82 members	Qt Belgium This group is for all Qt… 40 members
Qt China China Qt Developers 194 members	Qt Greece A group for Greek Qt users. 41 members	Qt USA A group for Qt developers in… 130 members	Qt Japan A group for Japanese members 45 members

Communities

Communities



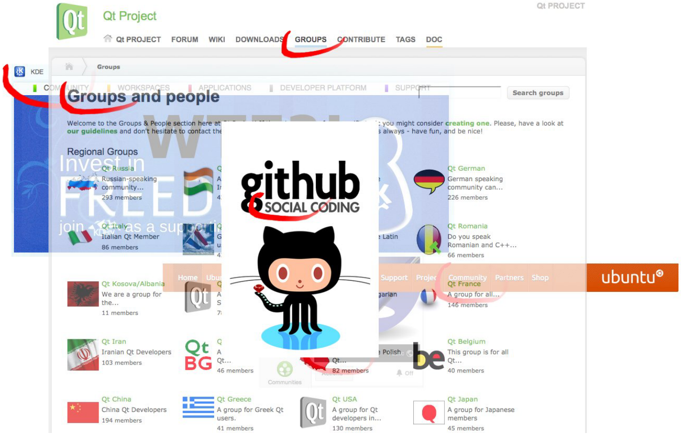
Groups and people

Welcome to the Groups & People section here at Qt Project! If there is no group for a specific topic you might consider **creating one**. Please, have a look at **our guidelines** and don't hesitate to contact the **admin team** with all questions and concerns. As always - have fun, and be nice!

Regional Groups

Qt Russia Russian-speaking community... 293 members	Qt India A group for the Indian Qt... 421 members	Qt Brazil A group for the Brazilian Qt... 161 members	Qt German German speaking community can... 226 members
Qt Italy Italian Qt Member 86 members	Qt Netherlands Group for Dutch Qt users. The... 43 members	Qt LatAm A group for the Latin America... 76 members	Qt Romania Do you speak Romanian and C++... 66 members
Qt Kosova/Albania We are a group for the... 11 members	Qt Spain A group of the Spanish... 78 members	Qt Hungary Group for Hungarian Qt users... 42 members	Qt France A group for all... 146 members
Qt Iran Iranian Qt Developers 103 members	Qt Bulgaria A group for Bulgarian Qt... 46 members	Qt Poland A group for the Polish Qt... 82 members	Qt Belgium This group is for all Qt... 40 members
Qt China China Qt Developers 194 members	Qt Greece A group for Greek Qt users. 41 members	Qt USA A group for Qt developers in... 130 members	Qt Japan A group for Japanese members 45 members

Communities



Communities (mine)

```
didier(s002)% ls /Applications \
               ‘echo $PATH | sed ’s/:/ /g’‘ 2>/dev/null \
              | wc -l

=> 3228
```

Communities (mine)

```
didier(s002)% ls /Applications \
               ‘echo $PATH | sed ’s/:/ /g’‘ 2>/dev/null \
              | wc -l

=> 3228



didier(s002)% expr 3228 \* 5 / 60

=> 269
```

A commune E.T.

Like

Lightning Talks

Pete Goodliffe – Design Sins
Alan Griffiths – The ACCU: Magic Happens Here
Dirk Haun – What's our Status?
Gail Ollis – Hello. I'm Back
Michel Grootjans – The Librarian
Björn Fahller – Why Are (only) We Here?
Frank Birbacher – Simple Quick Sort in C++
Calum Grant – C++ Active Objects
Didier Verna – Communities
Burkhard Kloss – Just a Minute
Guy Bolton King – BDD with Boost Test
Ed Sykes – A Decision Made With Data

accu 2013

Lightning Talks

Pete Goodliffe – Design Sins
Alan Griffiths – The ACCU: Magic Happens Here
Dirk Haun – What's our Status?
Gail Ollis – Hello. I'm Back
Michel Grootjans – The Librarian
Björn Fahller – Why Are (only) We Here?
Frank Birbacher – Simple Quick Sort in C++
Calum Grant – C++ Active Objects
Didier Verna – Communities
Burkhard Kloss – Just a Minute
Guy Bolton King – BDD with Boost Test
Ed Sykes – A Decision Made With Data

BDD with Boost Test

```
describe "a stack" do
  let(:stack) { [] }

  context "when initialised" do
    it "should be empty" do
      stack.should be_empty
    end
  end

  describe "pop" do
    context "on an empty stack" do
      before(:each) {
        stack.should be_empty
      }

      it "should have no effect" do
        stack.pop
        stack.should be_empty
      end
    end
                                   context "on a stack with a single member" do
                                     before(:each) {
                                       stack.push(1)
                                       stack.size.should be(1)
                                     }

                                     it "should result in an empty stack" do
                                       stack.pop
                                       stack.should be_empty
                                     end

                                     it "should reduce the stack size by one" do
                                       expect { stack.pop }.to change { stack.size }.by(-1)
                                     end
                                   end
                                 end
                               end
```

```
a stack
  when initialised
    should be empty
  pop
    on an empty stack
      should have no effect
    on a stack with a single member
      should result in an empty stack
      should reduce the stack size by one
```

```ruby
describe "a stack" do
  let(:stack) { [] }

  describe "pop" do
    context "on a stack with a single member" do
      before(:each) {
        stack.push(1)
        stack.size.should be(1)
      }

      it "should reduce the stack size by one" do
        expect {
          stack.pop
        }.to change { stack.size }.by(-1)
      end
    end
  end
end
```

```
struct a_stack_ {
  Stack<int> stack;
};

BOOST_FIXTURE_TEST_SUITE(a_stack, a_stack_)
  BOOST_AUTO_TEST_SUITE(pop)

    struct on_a_stack_with_a_single_member_: a_stack_ {
      on_a_stack_with_a_single_member_() {
        stack.push(1);
        BOOST_REQUIRE_EQUAL(stack.size(), 1);
      }
    };

    BOOST_FIXTURE_TEST_SUITE(on_a_stack_with_a_single_member,
                             on_a_stack_with_a_single_member_)
      BOOST_AUTO_TEST_CASE(should_reduce_the_stack_size_by_one)
      {
        std::size_t orig_size = stack.size();
        stack.pop();
        BOOST_CHECK_EQUAL(stack.size(), orig_size - 1);
      }
    BOOST_AUTO_TEST_SUITE_END()
  BOOST_AUTO_TEST_SUITE_END()
BOOST_AUTO_TEST_SUITE_END()
```

```
Entering test suite "a_stack"
Entering test suite "pop"
Entering test suite "on_a_stack_with_a_single_member"
Entering test case "should_reduce_the_stack_size_by_one"
Leaving test case "should_reduce_the_stack_size_by_one"
Leaving test suite "on_a_stack_with_a_single_member"
Leaving test suite "pop"
Leaving test suite "a_stack"
```

```
class SpecLogFormatter:
  public boost::unit_test::output::compiler_log_formatter {

public:
  SpecLogFormatter(): m_indent(0) {}

private:
  void test_unit_start(std::ostream &os,
    boost::unit_test::test_unit const& tu)
  {
    os << std::string(m_indent, ' ') <<
      boost::replace_all_copy(tu.p_name.get(), "_", " ") << std::endl;
    m_indent += 2;
  }

  void test_unit_finish(std::ostream &os,
    boost::unit_test::test_unit const& tu, unsigned long elapsed)
  {
    m_indent -= 2;
  }

  int m_indent;
};
```

```
a stack
  pop
    on a stack with a single member
      should reduce the stack size by one
```

```
a stack
  when initialised
    should be empty
  pop
    on an empty stack
      should have no effect
    on a stack with a single member
      should result in an empty stack
      should reduce the stack size by one
```

```
struct a_stack_ {
  Stack<int> stack;
};

BOOST_FIXTURE_TEST_SUITE(a_stack, a_stack_)
  BOOST_AUTO_TEST_SUITE(when_initialised)
    BOOST_AUTO_TEST_CASE(should_be_empty)
    {
      BOOST_CHECK(stack.empty());
    }
  BOOST_AUTO_TEST_SUITE_END()

  BOOST_AUTO_TEST_SUITE(pop)

    struct on_an_empty_stack_: a_stack_ {
      on_an_empty_stack_() {
        BOOST_REQUIRE(stack.empty());
      }
    };

    BOOST_FIXTURE_TEST_SUITE(on_an_empty_stack, on_an_empty_stack_)
      BOOST_AUTO_TEST_CASE(should_have_no_effect)
      {
        stack.pop();
        BOOST_CHECK(stack.empty());
      }
    BOOST_AUTO_TEST_SUITE_END()

    struct on_a_stack_with_a_single_member_: a_stack_ {
      on_a_stack_with_a_single_member_() {
        stack.push(1);
        BOOST_REQUIRE_EQUAL(stack.size(), 1);
      }
    };
    BOOST_FIXTURE_TEST_SUITE(on_a_stack_with_a_single_member,
                             on_a_stack_with_a_single_member_)
      BOOST_AUTO_TEST_CASE(should_result_in_an_empty_stack)
      {
        stack.pop();
        BOOST_CHECK(stack.empty());
      }

      BOOST_AUTO_TEST_CASE(should_reduce_the_stack_size_by_one)
      {
        std::size_t orig_size = stack.size();
        stack.pop();
        BOOST_CHECK_EQUAL(stack.size(), orig_size - 1);
      }
    BOOST_AUTO_TEST_SUITE_END()
  BOOST_AUTO_TEST_SUITE_END()
BOOST_AUTO_TEST_SUITE_END()
```

macros

Lightning Talks

Pete Goodliffe – Design Sins

Alan Griffiths – The ACCU: Magic Happens Here

Dirk Haun – What's our Status?

Gail Ollis – Hello. I'm Back

Michel Grootjans – The Librarian

Björn Fahller – Why Are (only) We Here?

Frank Birbacher – Simple Quick Sort in C++

Calum Grant – C++ Active Objects

Didier Verna – Communities

Burkhard Kloss – Just a Minute

Guy Bolton King – BDD with Boost Test

Ed Sykes – A Decision Made With Data