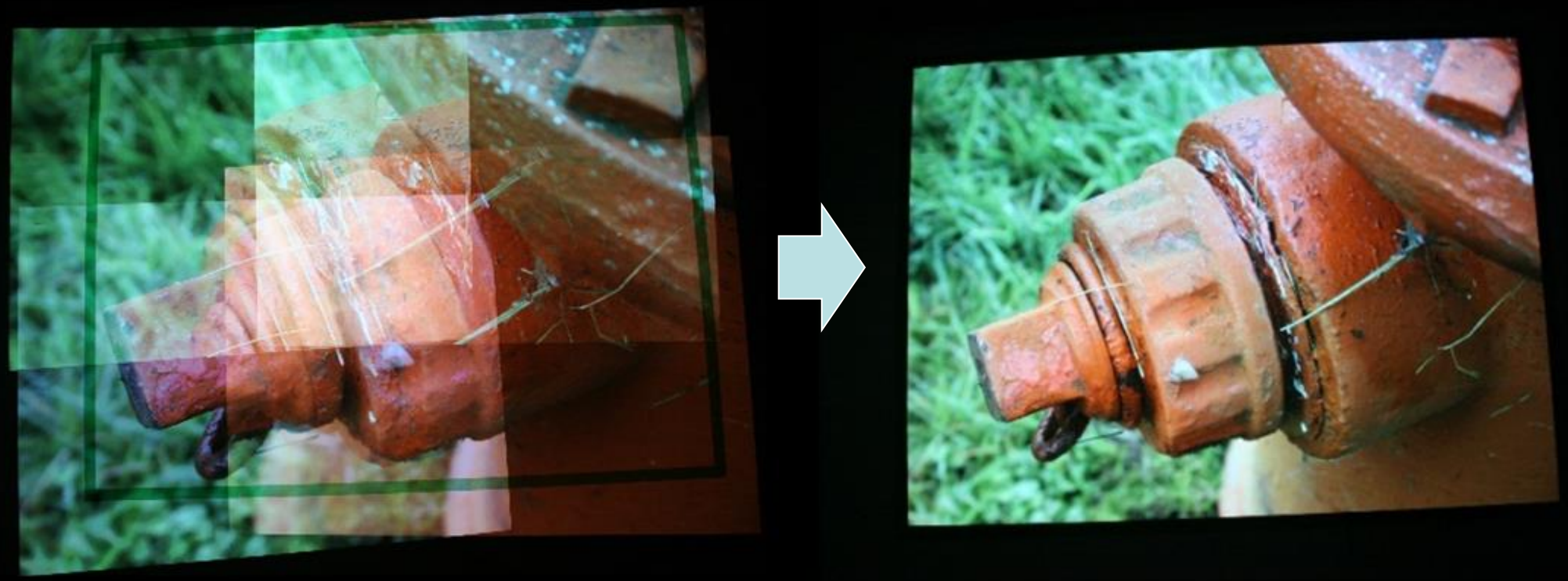
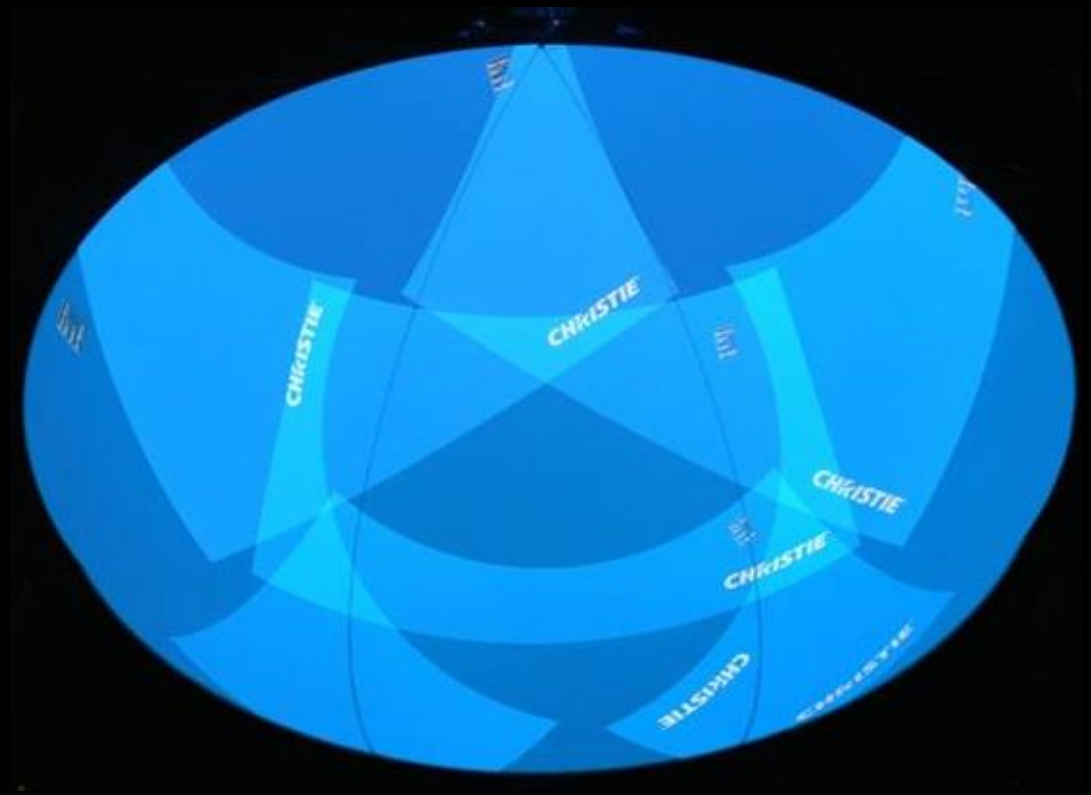
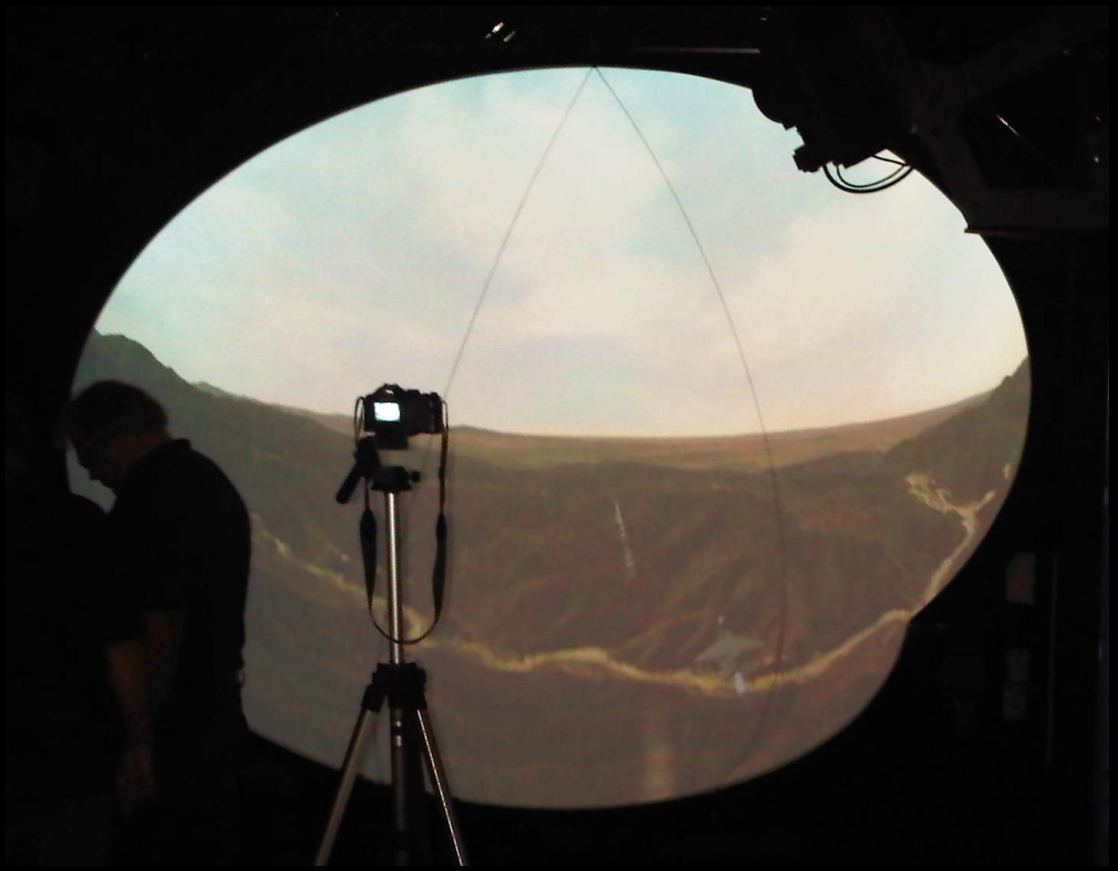

CHRISTIE®



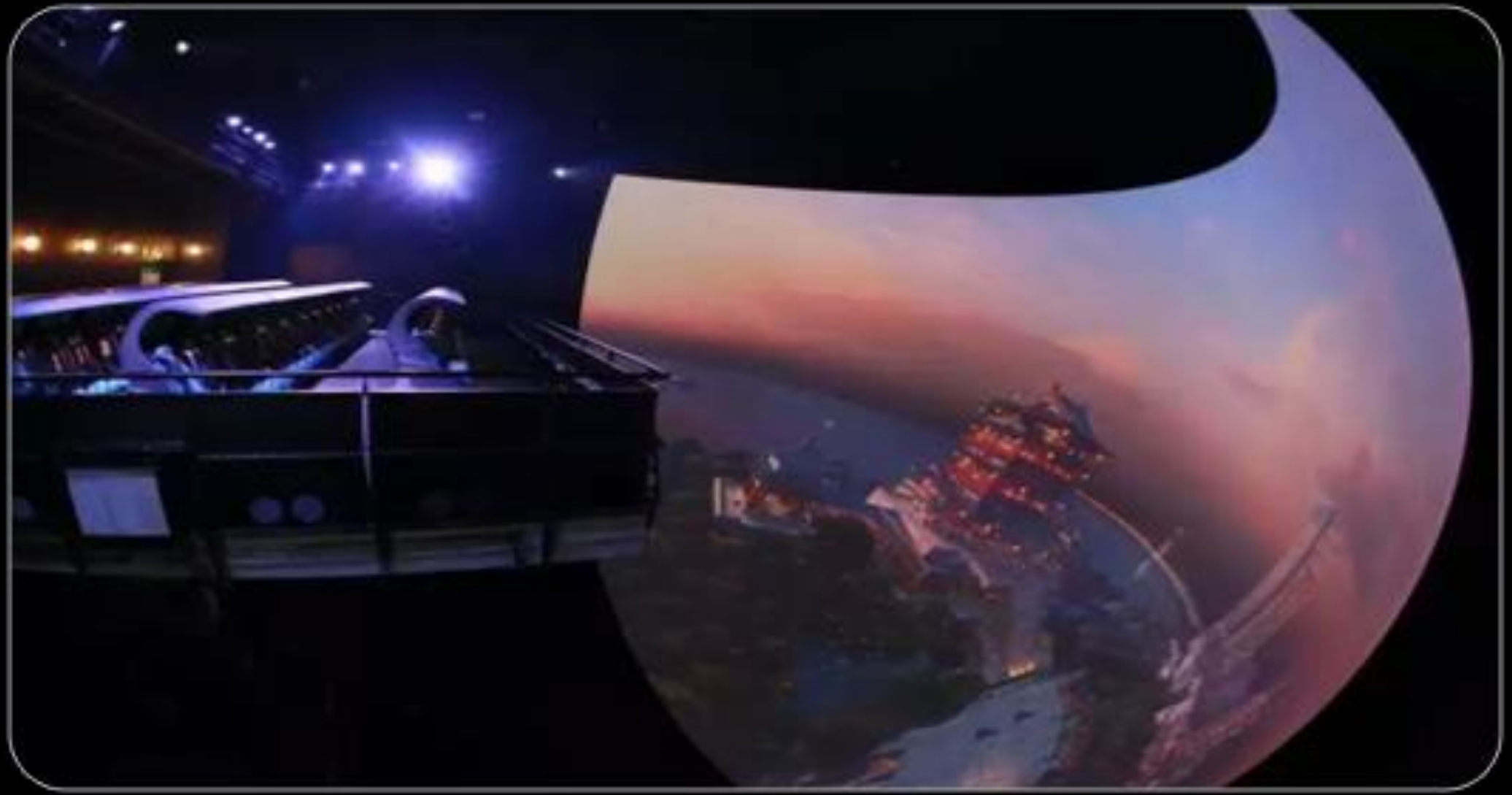




CHRISTIE®

CHRISTIE

CHRISTIE®



CHRISTIE


15
CENTER
CHRISTIE
CHRISTIE
CENTER
CHRISTIE
Christie Mystique
Spectacular experiences. Simple solutions.
Design
Install
Operate
BOEHME
Christie Sof
Superior ca
warping a
CHRISTIE

CHRISTIE

youtu.be/sql1kD6_Uok


New Seaside Dr
3D
Map
Google


T
3D
Map
Google

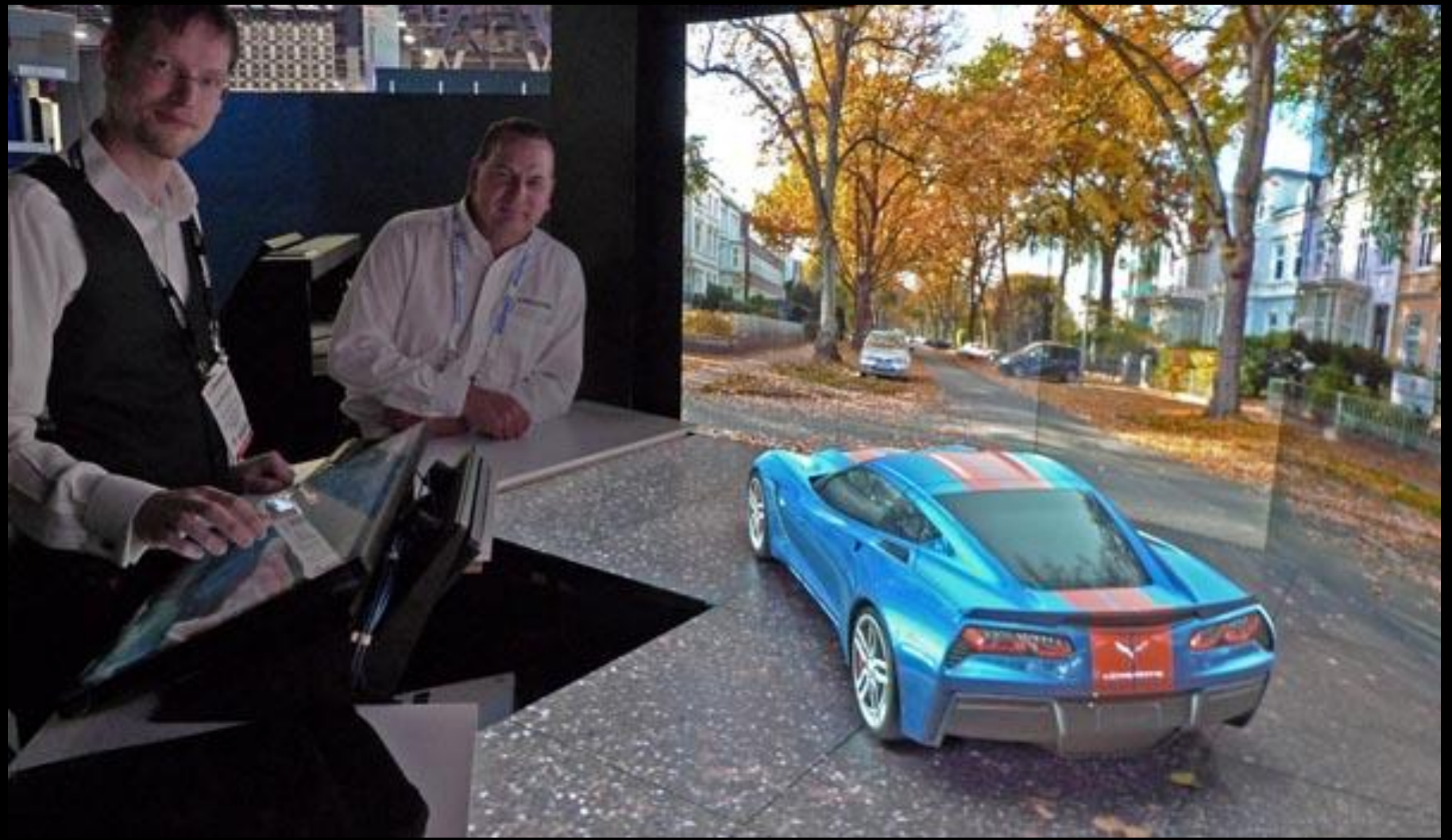
New Seaside Dr
3D
Map
Google

CHRISTIE

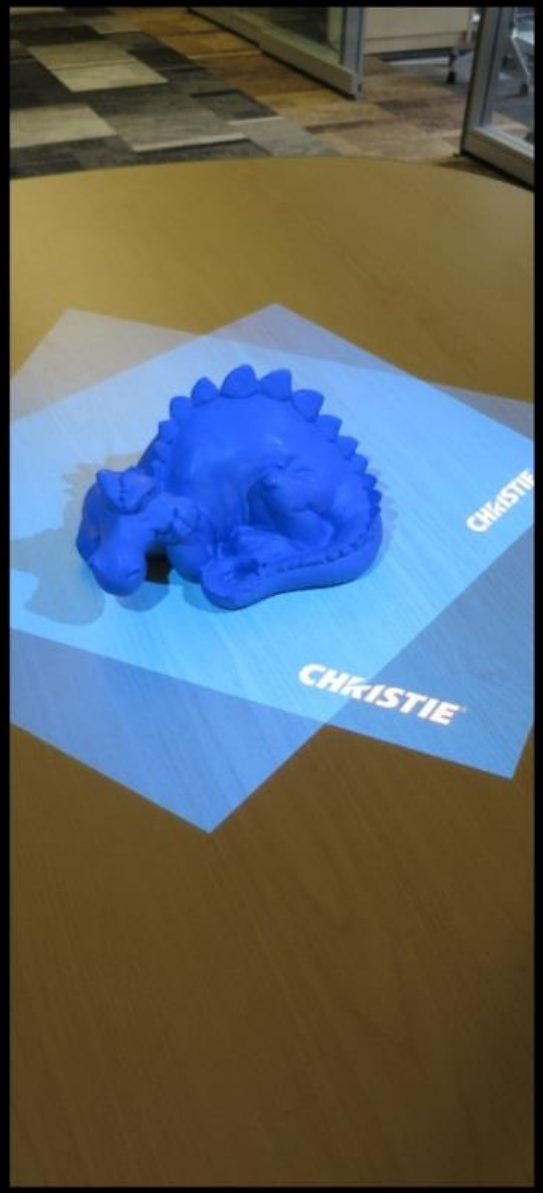
CHRISTIE

CHRISTIE

CHRISTIE



CHRISTIE


CHRISTIE
CHRISTIE

easy

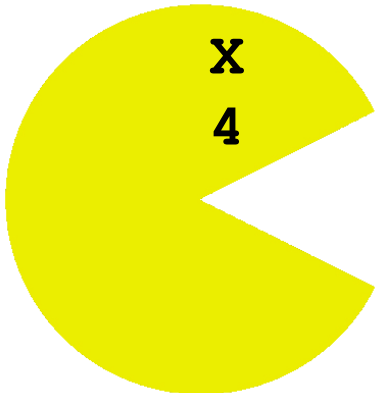

CHRISTIE®

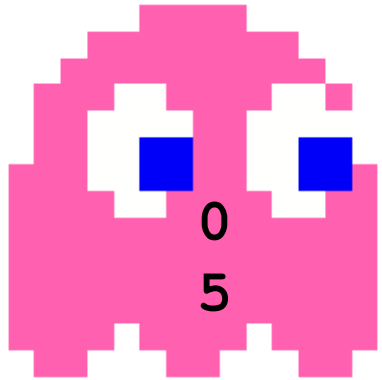
An *It's Getting Complicated Again* Lock-free Queue

Part 3 of N

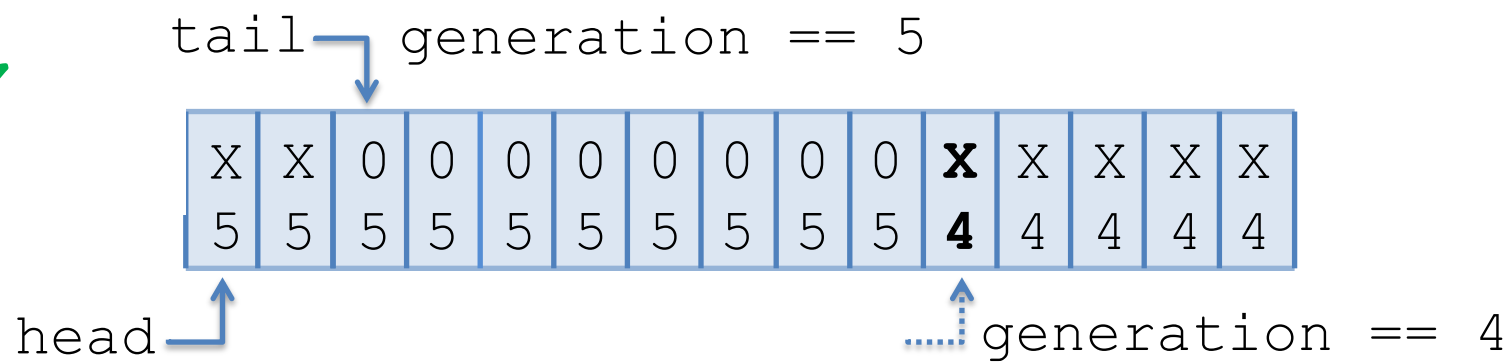
Tony Van Eerd C++Now May 2017

CHRISTIE






```
class Queue {
 atomic<entry> buffer[SIZE];
 laxtomic<int> head;
 laxtomic<int> h_generation;
 laxtomic<int> tail;
 laxtomic<int> t_generation;
};
```



Rules of Lock-free Programming

Rules of Lock-free Programming

1.

Rules of Lock-free Programming

1. Don’t talk about Lock-free Programming

Rules of Lock-free Programming

1. Don’t talk about Lock-free Programming
2.


Questions?

Rules of Lock-free Programming

1. Don’t talk about Lock-free Programming

Bonus Slides

Tony Van Eerd

C++Now 2017

Rules of Lock-free Programming

1. Don’t talk about Lock-free Programming

Rules of ~~Lock-free~~ Programming

1. Don’t talk about Lock-free Programming

Rules of ~~Lock-free~~ Programming

1. Don’t talk about Lock-free Programming
2. ***Use Locks***

Rules of ~~Lock-free~~ Programming

1. Don’t talk about Lock-free Programming
2. ***Use Locks***
3. MACROS ARE EVIL

Bonus Slides

Tony Van Eerd

C++Now 2017

Postmodern C++

Tony Van Eerd

C++Now 2017

Overview

Tony Van Eerd
@tvaneerd

How to "concentrate on one section of a programme at a time" without intrusion, and how to "test in isolation" here:

C++Now 2017: Postmodern C++
View more about this event at C++Now 2017
cppnow2017.sched.com

3:50 PM - 17 May 2017

Tony Van Eerd
@tvaneerd

Join me tomorrow at #CppNow - I will be *taking meta-programming to a whole new level* cppnow2017.sched.com/event/A8Iq/pos… CLICKBAIT: It will SHOCK YOU.

C++Now 2017: Postmodern C++
View more about this event at C++Now 2017
cppnow2017.sched.com

Tony Van Eerd
@tvaneerd

What _is_ #PostModernCpp? Find out at C++Now cppnow2017.sched.com/event/A8Iq/pos… CLICKBAIT: You'll never believe what a postmodern smart ptr looks like!

C++Now 2017: Postmodern C++
View more about this event at C++Now 2017
cppnow2017.sched.com

LIKES
2

Tony Van Eerd
@tvaneerd

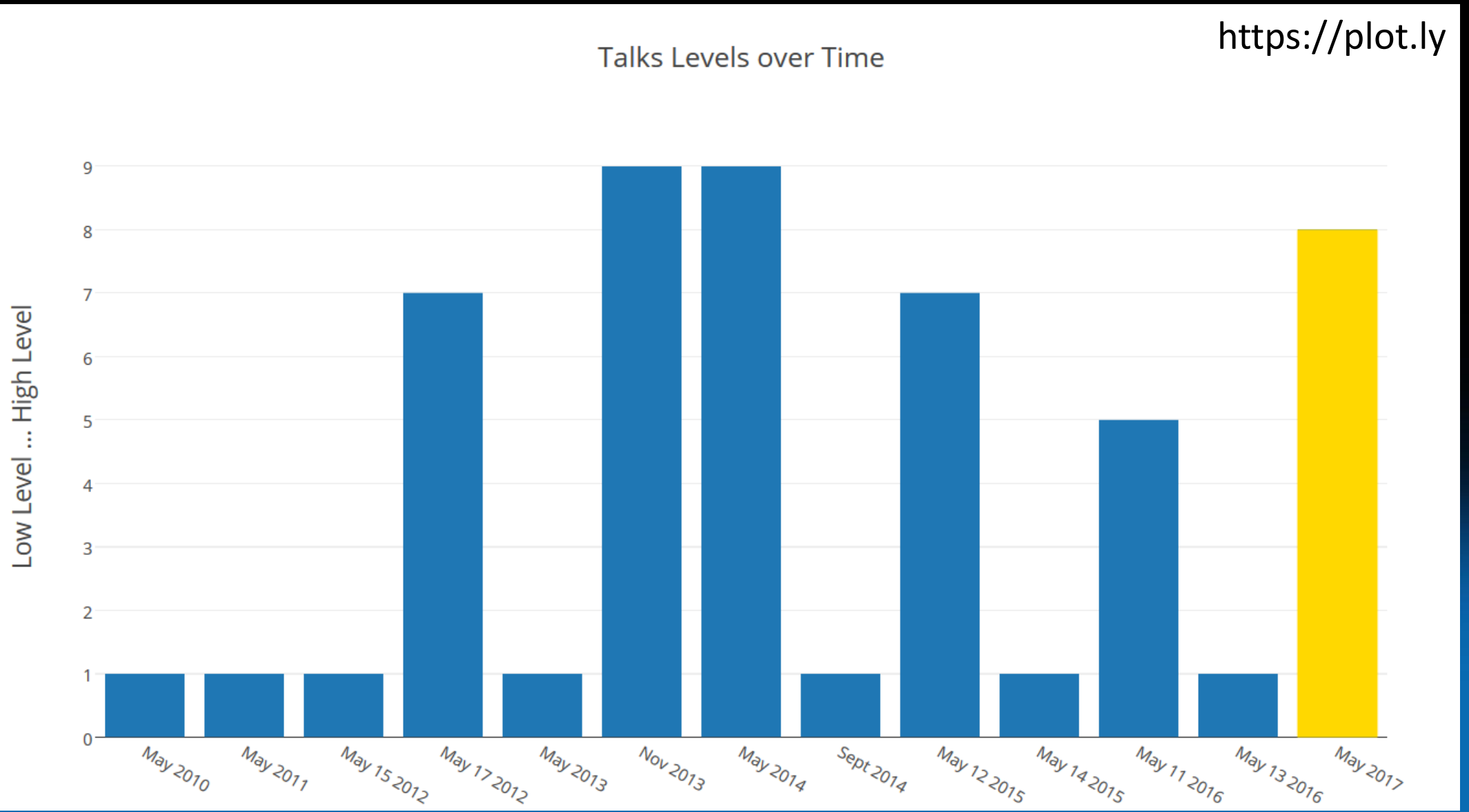
ULTIMATE CLICKBAIT: Later today I will perform _mind control_ on audience members at cppnow2017.sched.com/event/A8Iq/pos… YOU WON'T BELIEVE WHAT HAPPENS!

C++Now 2017: Postmodern C++
View more about this event at C++Now 2017
cppnow2017.sched.com

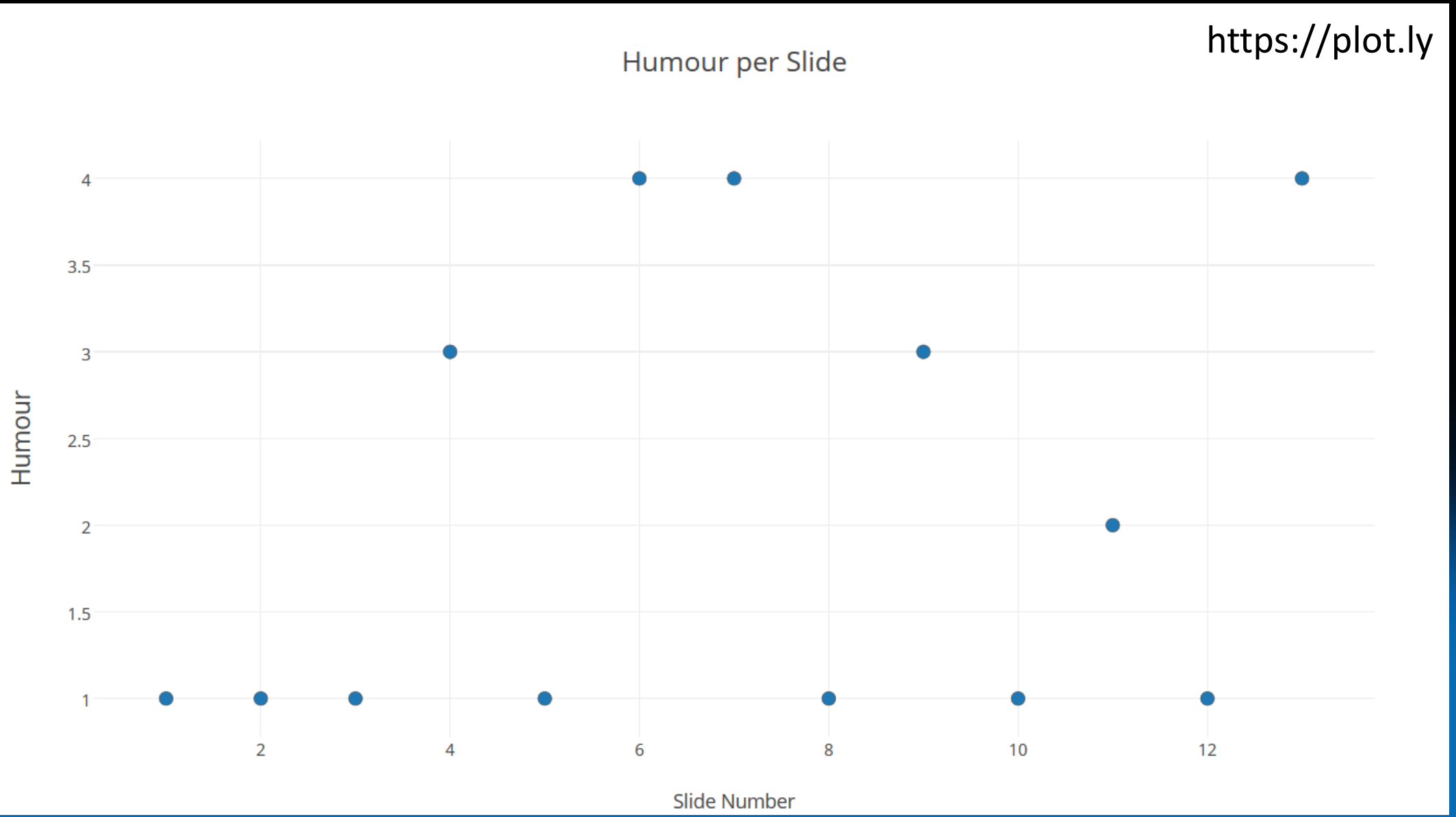
3:38 AM - 18 May 2017

I was told to have Graphs

https://plot.ly



https://plot.ly




Billy O'Neal @MalwareMinigun · 21 Sep 2016
I'm not an atomics expert. That class is approximately 2 weeks of my life :P
1
Meeting C++ @meetingcpp · 21 Sep 2016
you might want to ask Tony van Eerd for some feedback.
2
Tony Van Eerd
@tvaneerd
Follow
Replying to @meetingcpp @MalwareMinigun @jfbastien
Tony has been summoned and is now on Twitter. However nothing lockfree can be answered in 140 charac
RETWEETS
8
LIKES
9
10:37 AM - 21 Sep 2016
8
9

Meeting C++ @meetingcpp · 6 Oct 2016

Modern C++ is much older then 5 years, and has not much to do with the new standards...

2

Tony DaSilva @Bulldozer0 · 6 Oct 2016

??

1

Meeting C++ @meetingcpp · 6 Oct 2016

see my slides from CppCon, at the beginning (5th & 6th slide). With definition by @incomputable.

CppCon/CppCon2016

Slides and other materials from CppCon 2016. Contribute to CppCon2016 development by creating an account on GitHub.

github.com

2 1

Tony DaSilva @Bulldozer0 · 6 Oct 2016

Ah yes, now I remember. I think "modern" has taken on a whole new, broader scoped, meaning ever since C++11.

1

Meeting C++ @meetingcpp · 6 Oct 2016

I think its a really bad term to use today, as it means something different to so many people. @incomputable

scoped, meaning ever since C++11.

1

Meeting C++ @meetingcpp · 6 Oct 2016
I think its a really bad term to use today, as it means something different to so many people. @incomputable

2

Patrice Roy @PatriceRoy1 · 6 Oct 2016
Yeah; I think it came from @Scott__Meyers who asked readers guidance for his most recent book's title

2

Patrice Roy @PatriceRoy1 · 6 Oct 2016
I expect it to stick, though; «adapting to the new meaning» is probably wise

2

Meeting C++ @meetingcpp · 6 Oct 2016
which new meaning? There is such a variance in what it means to people...

3

Tony DaSilva @Bulldozer0 · 6 Oct 2016
I think the Cpp Core Guidelines are a good reference for seeing the "new modern" C++

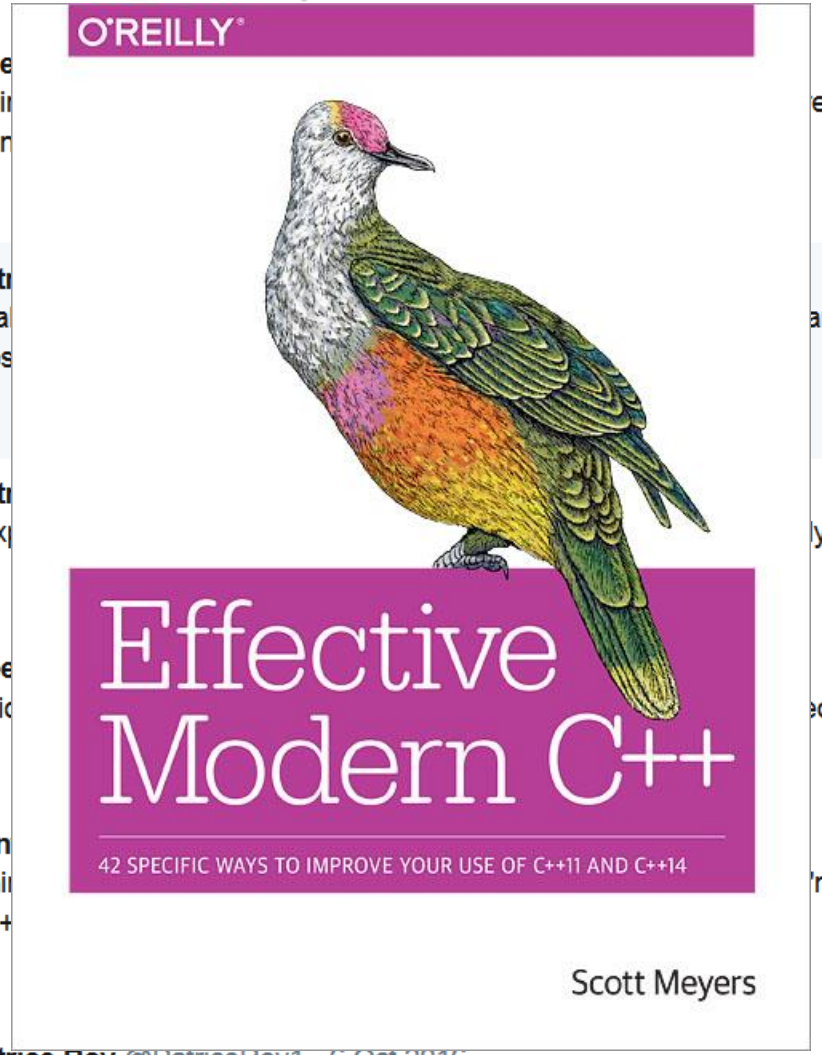
2 1

Patrice Roy @PatriceRoy1 · 6 Oct 2016
They're a bit large for that, IMHO, and I expect people to pick from it, not use it all

1

scoped, meaning ever since C++11.

1



Patrice Roy @PatriceRoy1 · 6 Oct 2016

They're a bit large for that, IMHO, and I expect people to pick from it, not use it all

1

scoped, meaning ever since C++11.

1

Meeting C++ @meetingcpp · 6 Oct 2016
I think its a really bad term to use today, as it means something different to so many people. @incomputable

2

Patrice Roy @PatriceRoy1 · 6 Oct 2016
Yeah; I think it came from @Scott__Meyers who asked readers guidance for his most recent book's title

2

Patrice Roy @PatriceRoy1 · 6 Oct 2016
I expect it to stick, though; «adapting to the new meaning» is probably wise

2

Meeting C++ @meetingcpp · 6 Oct 2016
which new meaning? There is such a variance in what it means to people...

3

Tony DaSilva @Bulldozer0 · 6 Oct 2016
I think the Cpp Core Guidelines are a good reference for seeing the "new modern" C++

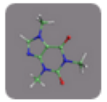
2 1

Patrice Roy @PatriceRoy1 · 6 Oct 2016
They're a bit large for that, IMHO, and I expect people to pick from it, not use it all

1

I think the Cpp Core Guidelines are a good reference for seeing the "new modern" C++

2 1



Patrice Roy @PatriceRoy1 · 6 Oct 2016

They're a bit large for that, IMHO, and I expect people to pick from it, not use it all

1

Tony DaSilva @Bulldozer0 · 6 Oct 2016

It's too large to fully use, but still lots of examples showing new "style" of coding?

1

Meeting C++ @meetingcpp · 6 Oct 2016

that "new style" was in use by boost for almost a decade earlier. Lots of people think that boost isn't modern...

1

Tony DaSilva @Bulldozer0 · 6 Oct 2016

Agreed, but not standardized until C++11

1

Meeting C++ @meetingcpp · 6 Oct 2016

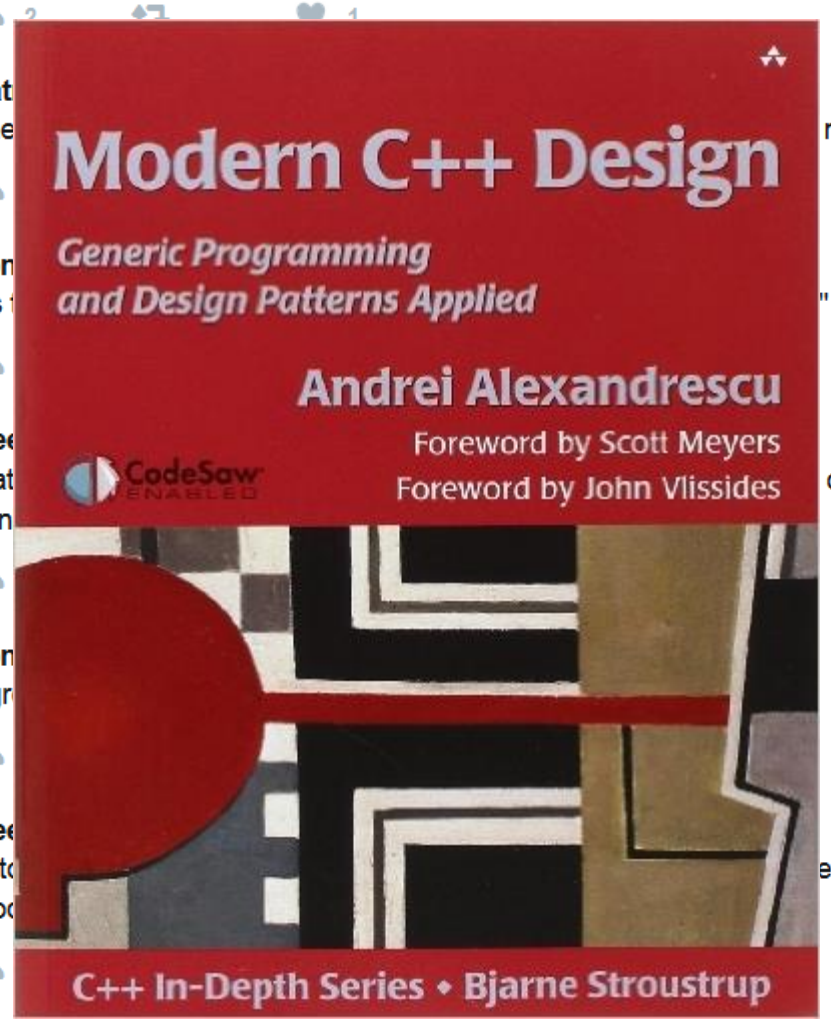
auto_ptr had slightly wrong semantics... I'm still sure the standard doesn't mention modern c++ ;)

2

Tony DaSilva @Bulldozer0 · 6 Oct 2016

Are we splitting hairs here? How would you distinguish pre-C++11 from post-C++11?

I think the Cpp Core Guidelines are a good reference for seeing the "new modern" C++



Tony DaSilva @Bulldozer0 · 6 Oct 2016

Are we splitting hairs here? How would you distinguish pre-C++11 from post-C++11?

Pearson Education Corporate Sales Division
201 W. 103rd Street
Indianapolis, IN 46290
(800) 428-5331
corpsales@pearsoned.com

Visit AW on the Web: www.awprofessional.com

Library of Congress Cataloging-in-Publication Data
Alexandrescu, Andrei.
Modern C++ design : generic programming and design patterns applied /
Andrei Alexandrescu.
p. cm. — (C++ in depth series)
Includes bibliographical references and index.
ISBN 0-201-70431-5
1. C++ (Computer program language) 2. Generic programming (Computer science)
I. Title. II. Series.
QA76.73.C153 A42 2001
005.13'3—dc21 00-049596

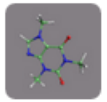
ISBN 0-201-70431-5

Text printed in the United States on recycled paper at RR Donnelley Crawfordsville in Crawfordsville, Indiana.

19th Printing January 2011

I think the Cpp Core Guidelines are a good reference for seeing the "new modern" C++

2 1



Patrice Roy @PatriceRoy1 · 6 Oct 2016

They're a bit large for that, IMHO, and I expect people to pick from it, not use it all

1

Tony DaSilva @Bulldozer0 · 6 Oct 2016

It's too large to fully use, but still lots of examples showing new "style" of coding?

1

Meeting C++ @meetingcpp · 6 Oct 2016

that "new style" was in use by boost for almost a decade earlier. Lots of people think that boost isn't modern...

1

Tony DaSilva @Bulldozer0 · 6 Oct 2016

Agreed, but not standardized until C++11

1

Meeting C++ @meetingcpp · 6 Oct 2016

auto_ptr had slightly wrong semantics... I'm still sure the standard doesn't mention modern c++ ;)

2

Tony DaSilva @Bulldozer0 · 6 Oct 2016

Are we splitting hairs here? How would you distinguish pre-C++11 from post-C++11?

post-C++11?

1

Meeting C++ @meetingcpp · 6 Oct 2016
how do you distinguish between the other standards? btw. don't use C++11, use C++14. or C++17. just saying ;)

3

Tony DaSilva @Bulldozer0 · 6 Oct 2016
C++11 was a large leap. 14 and (sadly) 17 are relatively minor.

1 1

Meeting C++ @meetingcpp · 6 Oct 2016
nope. Many new features and changes in both.

3

Tony DaSilva @Bulldozer0 · 6 Oct 2016
Sorry, but I disagree in the large. Features are relatively minor.

1 1

Meeting C++ @meetingcpp · 6 Oct 2016
sure, now you get to redefine modern C++ every decade, and people already abuse it as a buzzword. Things only will get worse...

1

Patrice Roy @PatriceRoy1 · 6 Oct 2016
modern. more-modern, more-more-modern... Works :)

Translate from German

1

post-C++11?

1

Meeting C++ @meetingcpp · 6 Oct 2016
how do you distinguish between the other standards? btw. don't use C++11, use C++14. or C++17. just saying ;)

3

Tony DaSilva @Bulldozer0 · 6 Oct 2016
C++11 was a large leap. 14 and (sadly) 17 are relatively minor.

1 1

Meeting C++ @meetingcpp · 6 Oct 2016
nope. Many new features and changes in both.

3

Tony DaSilva @Bulldozer0 · 6 Oct 2016
Sorry, but I disagree in the large. Features are relatively minor.

1 1

Meeting C++ @meetingcpp · 6 Oct 2016
sure, now you get to redefine modern C++ every decade, and people already abuse it as a buzzword. Things only will get worse...

1

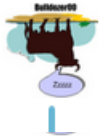
Patrice Roy @PatriceRoy1 · 6 Oct 2016
modern. more-modern, more-more-modern... Works :)

Translate from German

1

Meeting C++ @meetingcpp · 6 Oct 2016
auto_ptr had slightly wrong semantics... I'm still sure the standard doesn't mention modern c++ ;)

2

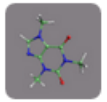


Tony DaSilva @Bulldozer0 · 6 Oct 2016
Are we splitting hairs here? How would you distinguish pre-C++11 from post-C++11?

1 1

Meeting C++ @meetingcpp · 6 Oct 2016
sure, now you get to redefine modern C++ every decade, and people already abuse it as a buzzword. Things only will get worse...

1



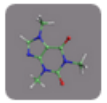
Patrice Roy @PatriceRoy1 · 6 Oct 2016
modern. more-modern, more-more-modern... Works :)

Translate from German

1

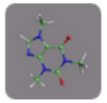
Tony Van Eerd @tvaneerd · 6 Oct 2016
post-modern, obviously. With Scott retiring from C++, I claim "Post-modern C++" as my book title. mine

3 3 12



Patrice Roy @PatriceRoy1 · 6 Oct 2016
If it's anything like other postmodernist movements, I definitely will want to read your book :)

1



Patrice Roy @PatriceRoy1 · 6 Oct 2016

modern. more-modern, more-more-modern... Works :)

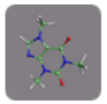
Translate from German

1

Tony Van Eerd @tvaneerd · 6 Oct 2016

post-modern, obviously. With Scott retiring from C++, I claim "Post-modern C++" as my book title. mine

3 3 12



Patrice Roy @PatriceRoy1 · 6 Oct 2016

If it's anything like other postmodernist movements, I definitely will want to read your book :)

1

Tony Van Eerd

@tvaneerd

Replying to @PatriceRoy1 @meetingcpp @Bulldozer0

It will have to be postmodern. The title forces the commitment. There's no going back now.

LIKE
1

C++

7:58 AM - 6 Oct 2016

Anyone here understand

Postmodernism?

Anyone here understand

Postmodernism?

Please leave.

Anyone here understand

Postmodernism?

Please leave. :-)

Tony Van Eerd
@tvaneerd

The meaning of a Container is in the _différance_ it has with the Containers that it is not. #PostModernC++. Except vector. Just use vector.

RETWEETS 2 LIKES 2

10:31 AM - 14 Oct 2016

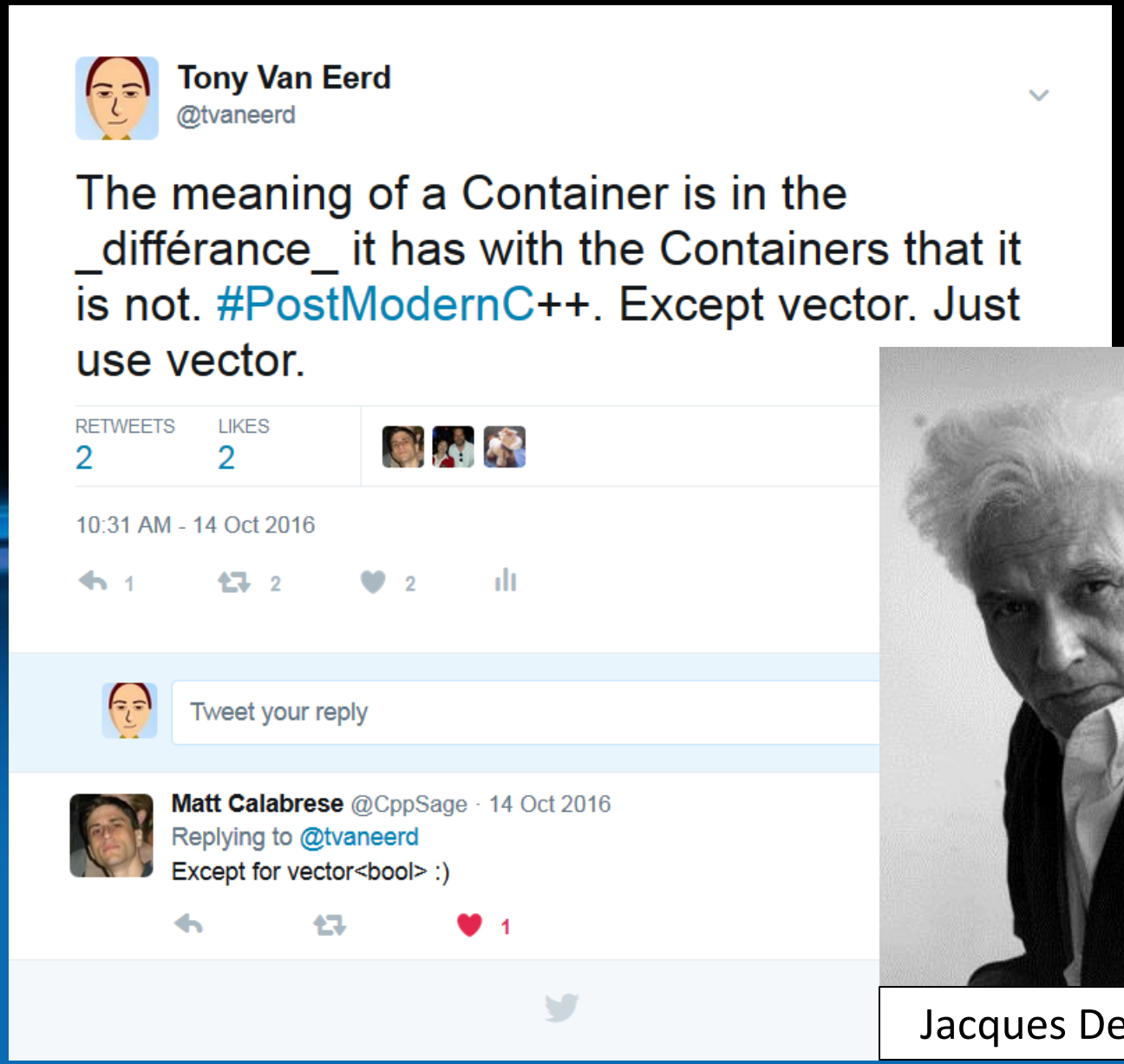
1 2 2

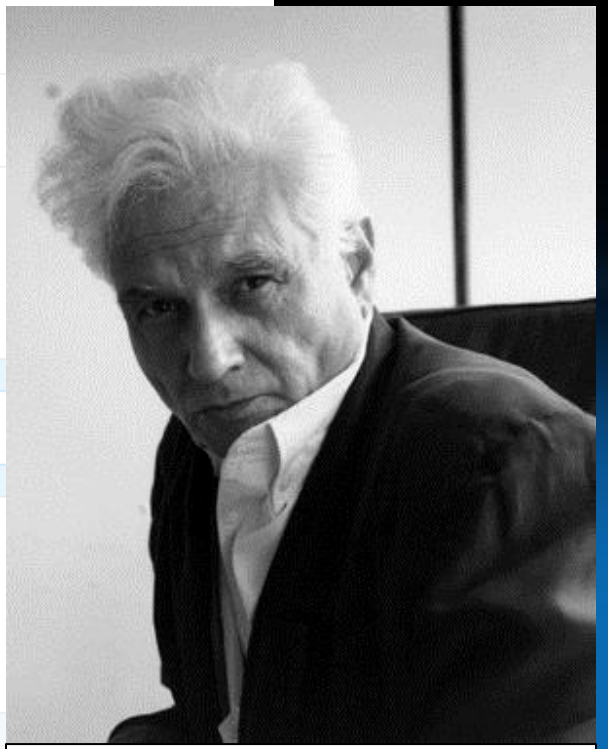
Tweet your reply

Matt Calabrese @CppSage · 14 Oct 2016
Replying to @tvaneerd
Except for vector<bool> :)

1

“Différance”

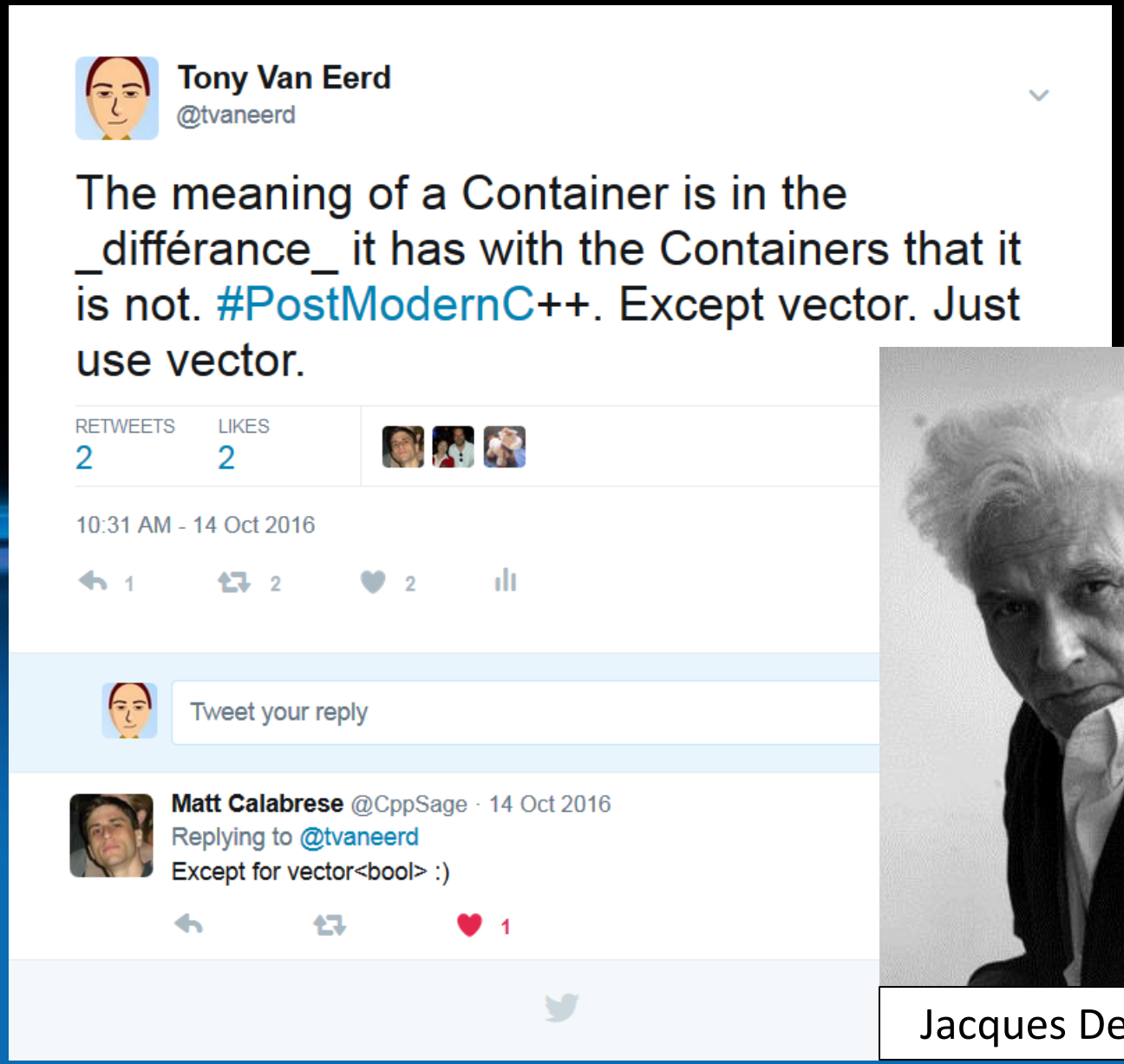


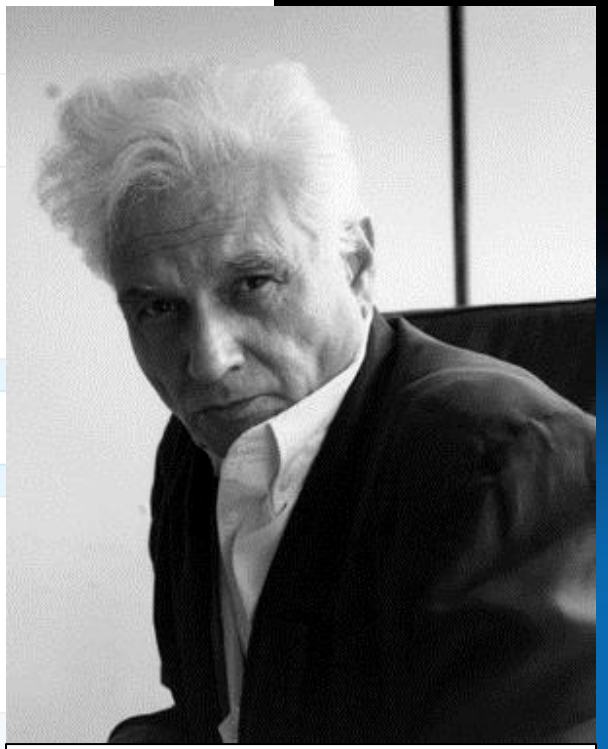


Jacques Derrida 1930-2004

Edna Kenny
(Resigning) Taoiseach
(Prime Minister) of Ireland

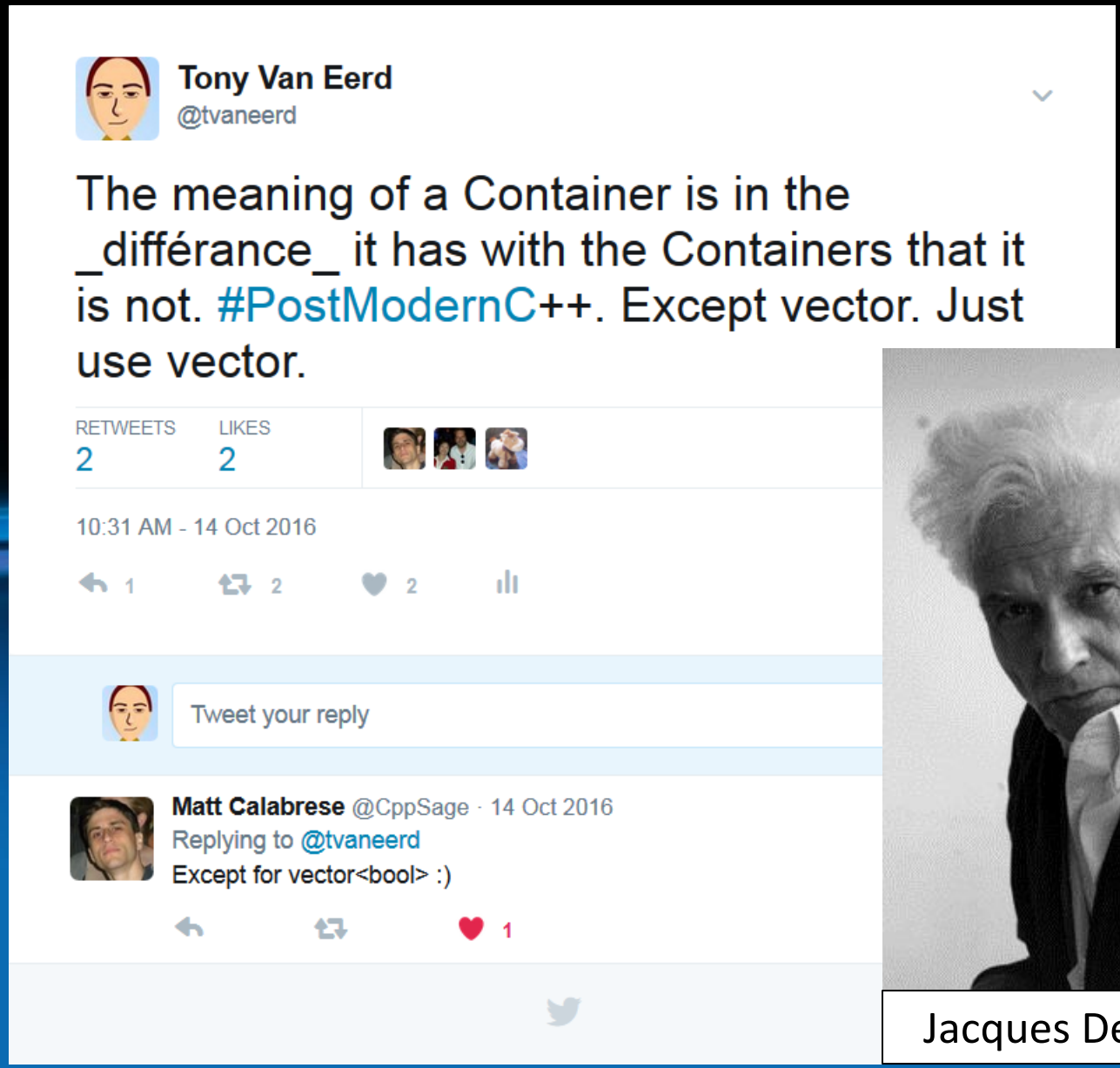
“Différance”

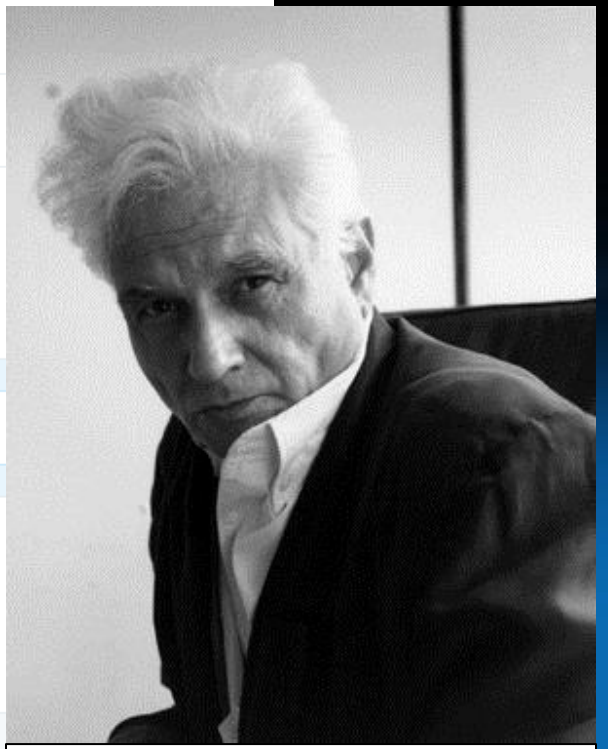




Jacques Derrida 1930-2004

- vector vs list, etc
- “real life”
- ConnectedImage
- Différance
- Deconstruction
- Architecture
- unsigned float



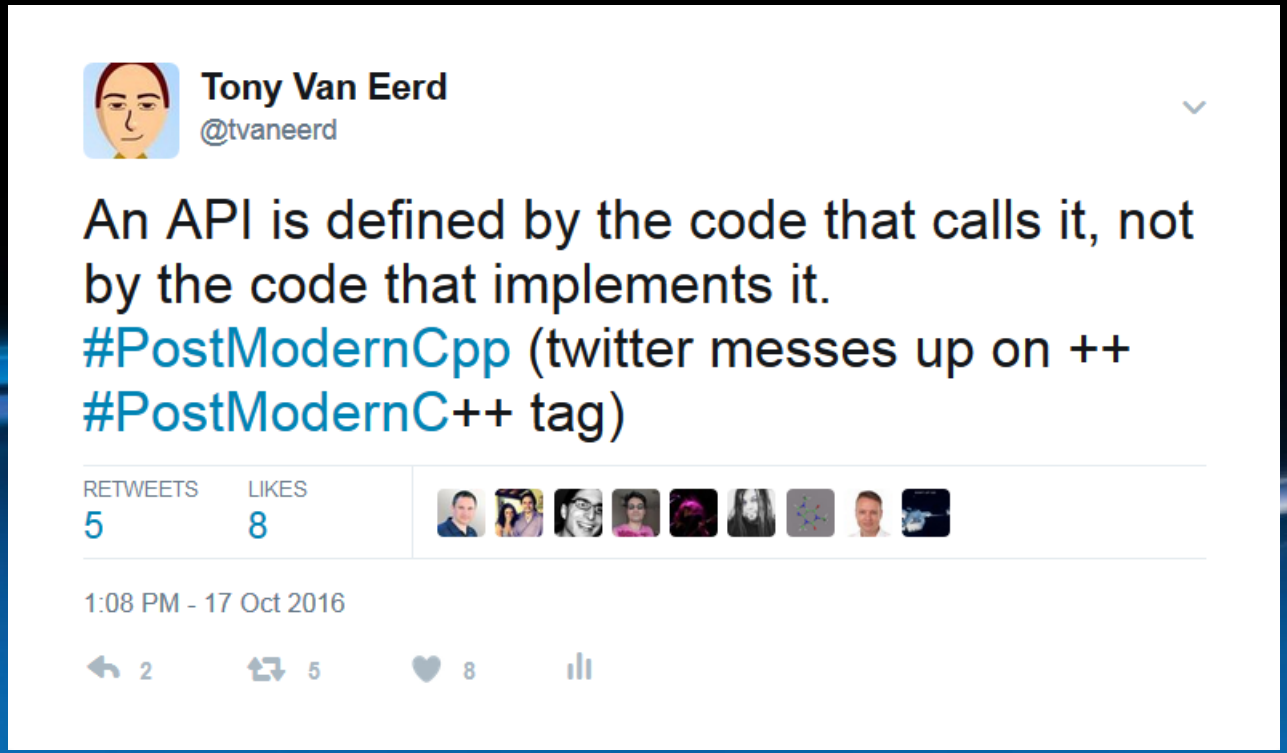


Jacques Derrida 1930-2004

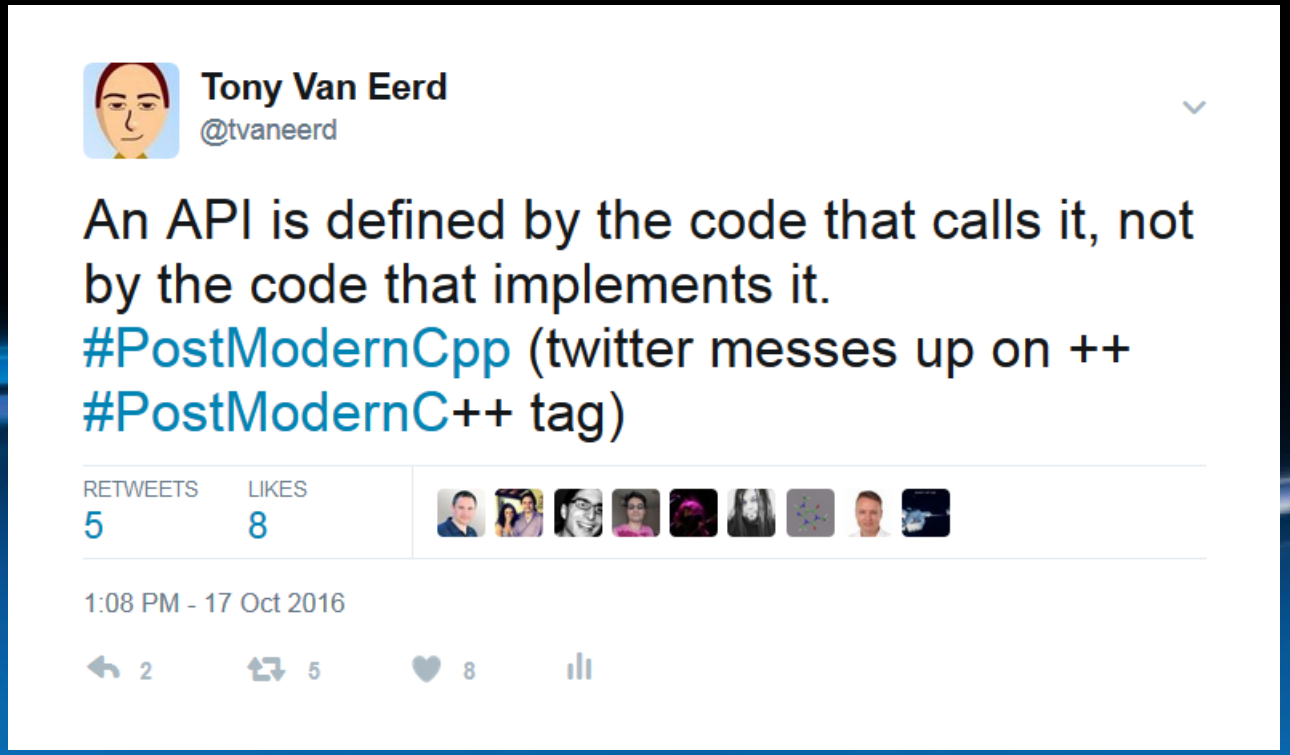

Tony Van Eerd
@tvaneerd
An API is defined by the code that calls it, not by the code that implements it. #PostModernCpp (twitter messes up on ++ #PostModernC++ tag)
RETWEETS
5
LIKES
8
1:08 PM - 17 Oct 2016
2
5
8

“PostModernC? You’d need Modern C first.”

- JF Bastien

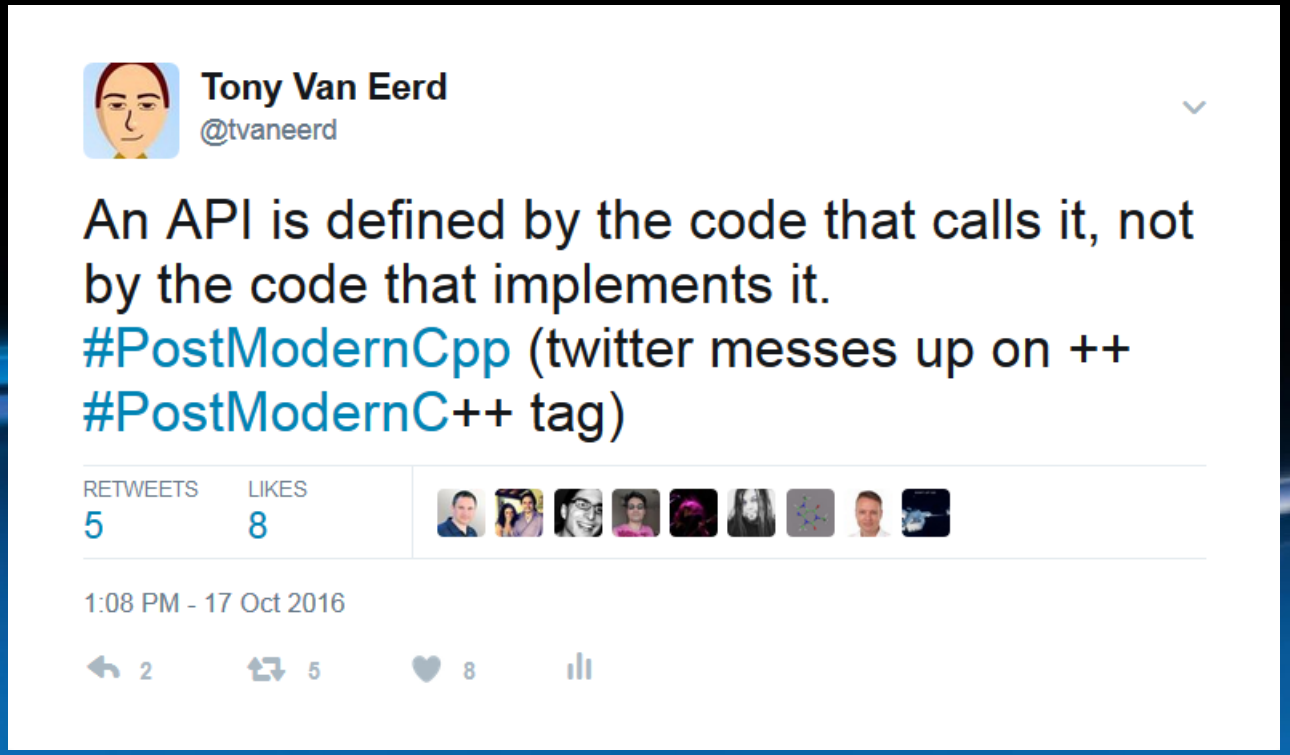


- deconstruction
- insight?



- deconstruction
- insight?

- Usage patterns
- Test/Usage first
- Indebted to users



Tony Van Eerd
@tvaneerd

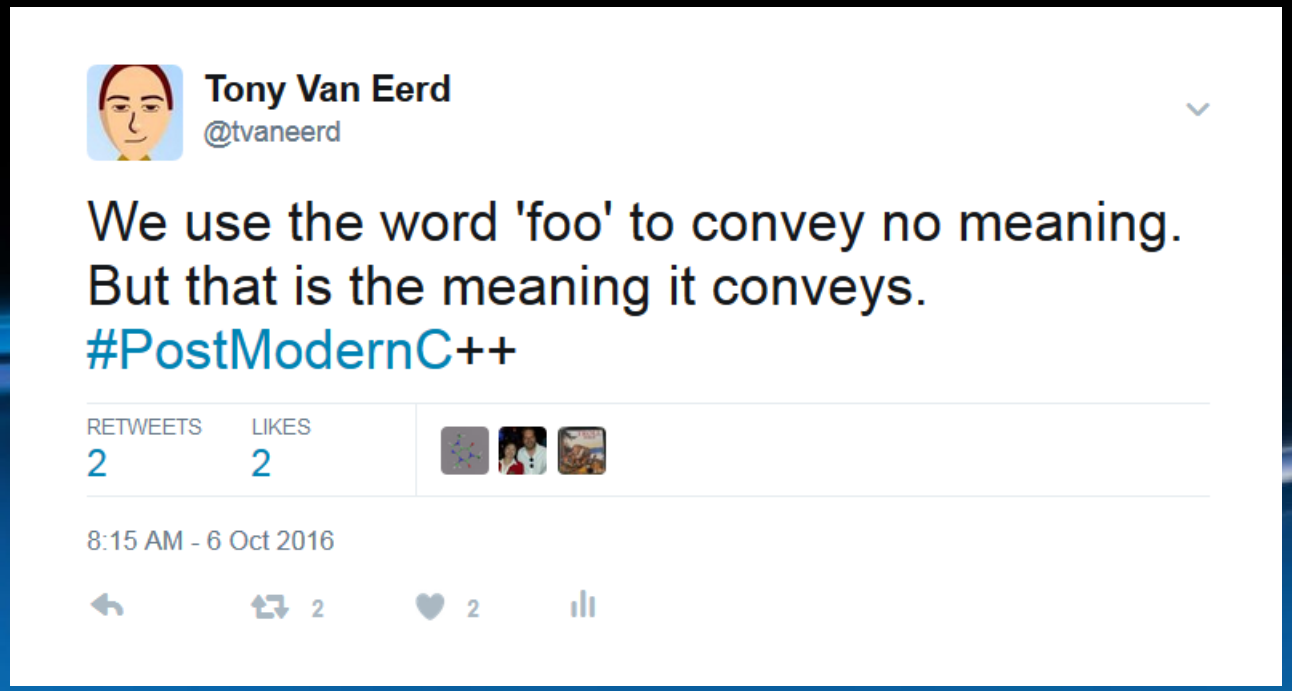
We use the word 'foo' to convey no meaning.
But that is the meaning it conveys.
#PostModernC++

RETWEETS	LIKES
2	2

8:15 AM - 6 Oct 2016

2 2

Irony.
Dispair.
Ultimate defferal
and différance



Tony Van Eerd
@tvaneerd

You must consider bracket styles within the context of the social/economic society in which they were formed. ie screen size. #PostModernC++

LIKES
2

11:48 AM - 7 Oct 2016

1 2

Tweet your reply

Michael Caisse @MichaelCaisse · 9 Oct 2016
Replying to @tvaneerd
My bracket style changed over the years with the social/economic status climbs of my equipment.

2

How To Deconstruct Almost Anything

My Postmodern Adventure

by Chip Morningstar

June 1993

(http://www.fudco.com/chip/deconstr.html)

"The essential paradigm of cyberspace is creating partially situated identities out of actual or potential social reality in terms of canonical forms of human contact, thus renormalizing the phenomenology of narrative space and requiring the naturalization of the intersubjective cognitive strategy, and thereby resolving the dialectics of metaphorical thoughts, each problematic to the other, collectively redefining and reifying the paradigm of the parable of the model of the metaphor."

ie
- avatars
- Twitter
- youtube comments?

How To Deconstruct Almost Anything

My Postmodern Adventure

by Chip Morningstar

June 1993

(http://www.fudco.com/chip/deconstr.html)

"The essential paradigm of cyberspace is creating partially situated identities out of actual or potential social reality in terms of canonical forms of human contact, thus renormalizing the phenomenology of narrative space and requiring the naturalization of the intersubjective cognitive strategy, and thereby resolving the dialectics of metaphorical thoughts, each problematic to the other, collectively redefining and reifying the paradigm of the parable of the model of the metaphor."

Tony Van Eerd
@tvaneerd

You must consider bracket styles within the context of the social/economic society in which they were formed. ie screen size. #PostModernC++

LIKES
2

11:48 AM - 7 Oct 2016

1 2

Tweet your reply

Michael Caisse @MichaelCaisse · 9 Oct 2016
Replying to @tvaneerd
My bracket style changed over the years with the social/economic status climbs of my equipment.

2

Tony Van Eerd
@tvaneerd

How we write is as important as what we write. If-statement on a single line? Compiler should translate that into __unlikely. #PostModernCpp

RETWEETS 2 LIKES 5

2:09 PM - 24 Oct 2016

1 2 5

Tweet your reply

JF Bastien @jfbastien · 24 Oct 2016
Replying to @tvaneerd
if condition contains the work "error" or "exception"? That's a hint for __unlikely!
#SpecManShip

Tony Van Eerd
@tvaneerd

How we write is as important as what we write. If-statement on a single line? Compiler should translate that into __unlikely. #PostModernCpp

RETWEETS 2 LIKES 5

🃏 Jest-in-Time compiler 🃏 LLVM wrangler 🃏 WebAssembly co-instigator 🃏 JavaScript fast-maker 🃏 C++ standards committee punster 🃏 🈸 🃏

JF Bastien @jfbastien · 24 Oct 2016
Replying to @tvaneerd
if condition contains the work "error" or "exception"? That's a hint for __unlikely!
#SpecManShip

```
if (condition) statement;

if (condition)
    statement;

if (condition) {
    statement;
}

if (condition)
{
    statement;
}
```

```
if (condition) statement;
```

Breakpoint?

```
if (condition)
   statement;

if (condition) {
   statement;
}

if (condition)
{
   statement;
}
```

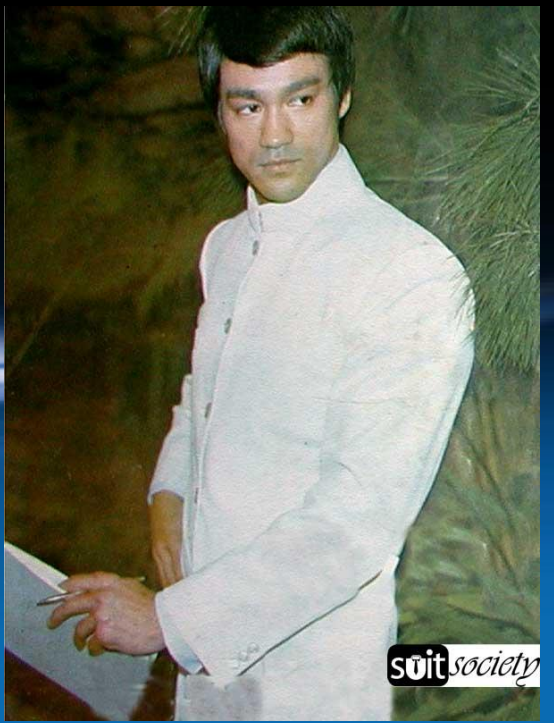
```
if (condition) statement;

if (condition)
    statement;

if (condition) {
    statement;
}

if (condition)
{
    statement;
}
```

Breakpoint?

```
if (condition)
    statement;
    statement;
```

“When one has no form, one can be all forms;
when one has no style, he can fit in with any style.”

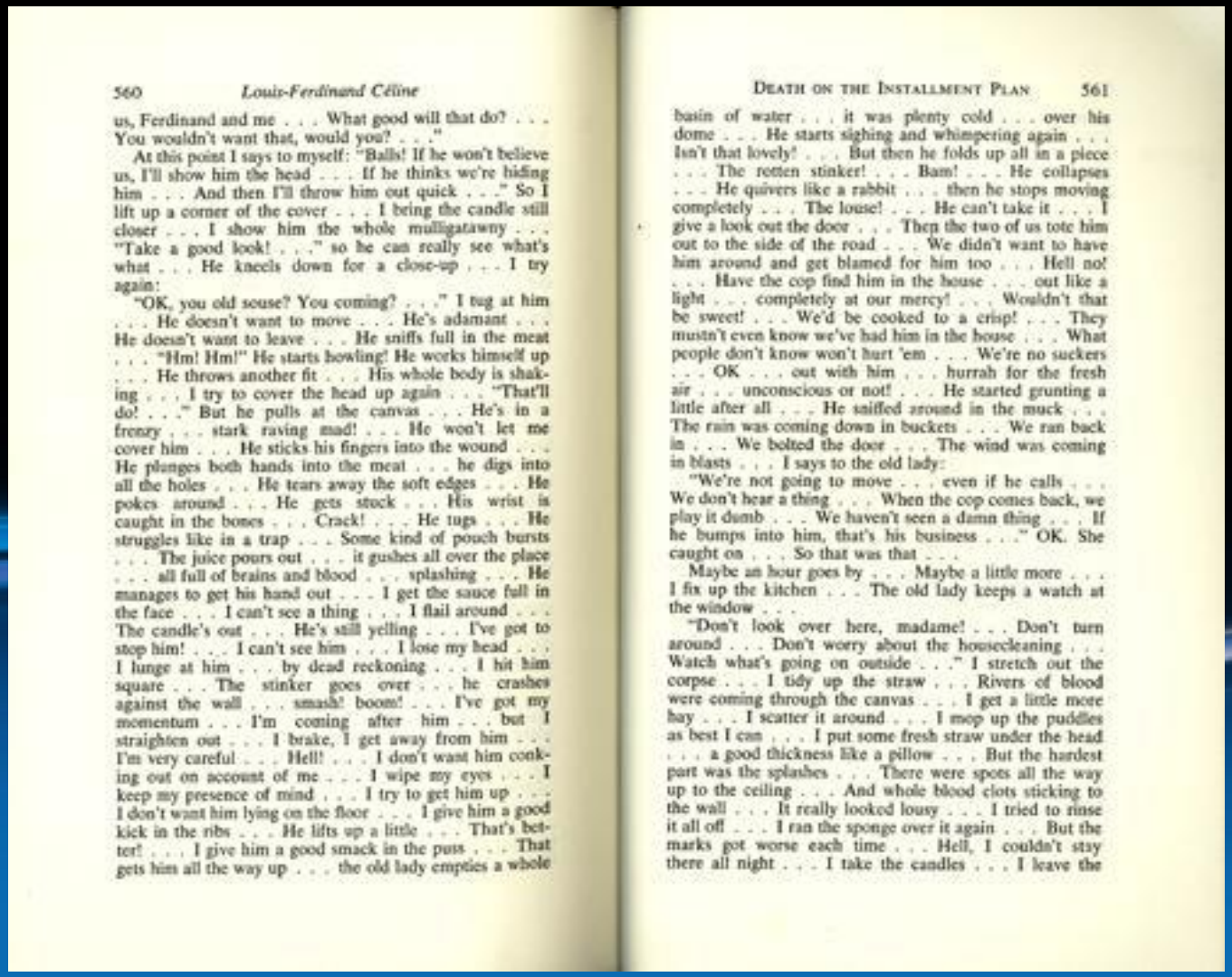
- Bruce Lee



```
if (condition) statement;

if (condition)
    statement;

if (condition) {
    statement;
}

if (condition)
{
    statement;
    statement;
}
```

- Significance
- Use the power of formatting for good

```
if (condition) statement;

if (condition)
    statement;

if (condition) {
    statement;
}

if (condition)
{
    statement;
    statement;
}
```



560 *Louis-Ferdinand Céline*

us, Ferdinand and me . . . What good will that do? . . . You wouldn't want that, would you? . . ."

At this point I says to myself: "Balls! If he won't believe us, I'll show him the head . . . If he thinks we're hiding him . . . And then I'll throw him out quick . . ." So I lift up a corner of the cover . . . I bring the candle still closer . . . I show him the whole mulligatawny . . . "Take a good look! . . ." so he can really see what's what . . . He kneels down for a close-up . . . I try again:

"OK, you old souse? You coming? . . ." I tug at him . . . He doesn't want to move . . . He's adamant . . . He doesn't want to leave . . . He sniffs full in the meat . . . "Hm! Hm!" He starts howling! He works himself up . . . He throws another fit . . . His whole body is shaking . . . I try to cover the head up again . . . "That'll do! . . ." But he pulls at the canvas . . . He's in a frenzy . . . stark raving mad! . . . He won't let me cover him . . . He sticks his fingers into the wound . . . He plunges both hands into the meat . . . he digs into all the holes . . . He tears away the soft edges . . . He pokes around . . . He gets stuck . . . His wrist is caught in the bones . . . Crack! . . . He tugs . . . He struggles like in a trap . . . Some kind of pouch bursts . . . The juice pours out . . . it gushes all over the place . . . all full of brains and blood . . . splashing . . . He manages to get his hand out . . . I get the sauce full in the face . . . I can't see a thing . . . I flail around . . . The candle's out . . . He's still yelling . . . I've got to stop him! . . . I can't see him . . . I lose my head . . . I lunge at him . . . by dead reckoning . . . I hit him square . . . The stinker goes over . . . he crashes against the wall . . . smash! boom! . . . I've got my momentum . . . I'm coming after him . . . but I straighten out . . . I brake, I get away from him . . . I'm very careful . . . Hell! . . . I don't want him conking out on account of me . . . I wipe my eyes . . . I keep my presence of mind . . . I try to get him up . . . I don't want him lying on the floor . . . I give him a good kick in the ribs . . . He lifts up a little . . . That's better! . . . I give him a good smack in the puss . . . That gets him all the way up . . . the old lady empties a whole

DEATH ON THE INSTALLMENT PLAN 561

basin of water . . . it was plenty cold . . . over his dome . . . He starts sighing and whimpering again . . . Isn't that lovely! . . . But then he folds up all in a piece . . . The rotten stinker! . . . Bam! . . . He collapses . . . He quivers like a rabbit . . . then he stops moving completely . . . The louse! . . . He can't take it . . . I give a look out the door . . . Then the two of us tote him out to the side of the road . . . We didn't want to have him around and get blamed for him too . . . Hell no! . . . Have the cop find him in the house . . . out like a light . . . completely at our mercy! . . . Wouldn't that be sweet! . . . We'd be cooked to a crisp! . . . They mustn't even know we've had him in the house . . . What people don't know won't hurt 'em . . . We're no suckers . . . OK . . . out with him . . . hurrah for the fresh air . . . unconscious or not! . . . He started grunting a little after all . . . He sniffed around in the muck . . . The rain was coming down in buckets . . . We ran back in . . . We bolted the door . . . The wind was coming in blasts . . . I says to the old lady:

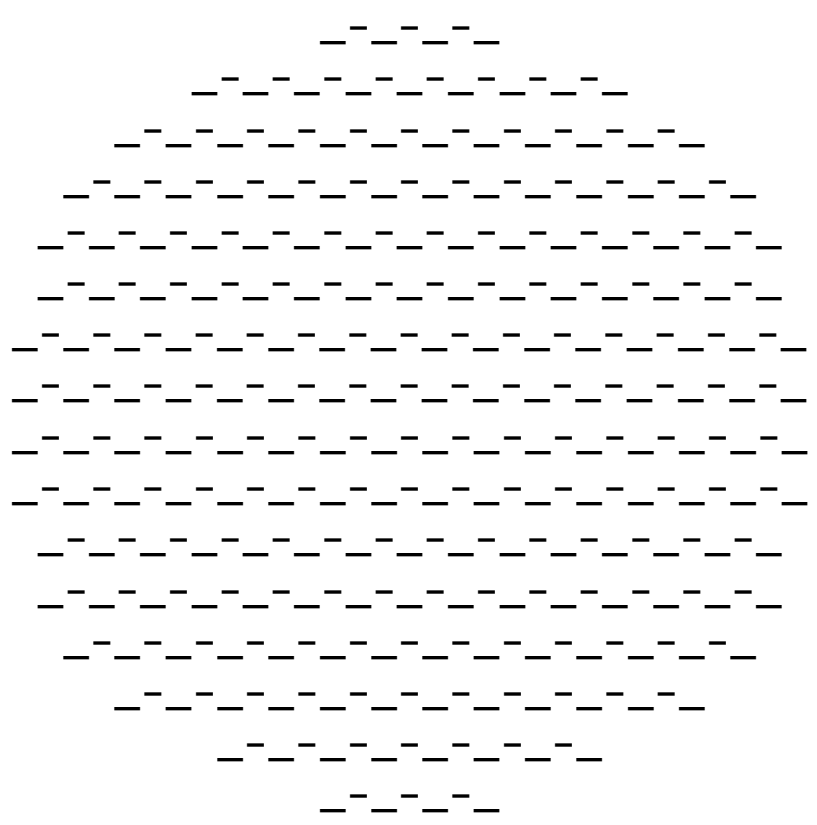
"We're not going to move . . . even if he calls . . . We don't hear a thing . . . When the cop comes back, we play it dumb . . . We haven't seen a damn thing . . ." OK. She caught on . . . So that was that . . .

Maybe an hour goes by . . . Maybe a little more . . . I fix up the kitchen . . . The old lady keeps a watch at the window . . .

"Don't look over here, madame! . . . Don't turn around . . . Don't worry about the housecleaning . . . Watch what's going on outside . . ." I stretch out the corpse . . . I tidy up the straw . . . Rivers of blood were coming through the canvas . . . I get a little more hay . . . I scatter it around . . . I mop up the puddles as best I can . . . I put some fresh straw under the head . . . a good thickness like a pillow . . . But the hardest part was the splashes . . . There were spots all the way up to the ceiling . . . And whole blood clots sticking to the wall . . . It really looked lousy . . . I tried to rinse it all off . . . I ran the sponge over it again . . . But the marks got worse each time . . . Hell, I couldn't stay there all night . . . I take the candles . . . I leave the

Louis-Ferdinand Céline, *Mort à Crédit (Death on the Installment Plan)*. (1936)

```
#define _ F-->00||-F-OO--;
int F=00,OO=00;main(){F_OO();printf("%1.3f\n",4.*-F/OO/OO);}F_OO()
{
```



```
}
```

```
// by
// Brian Westley
// aka Merlyn LeRoy
```

Tony Van Eerd
@tvaneerd

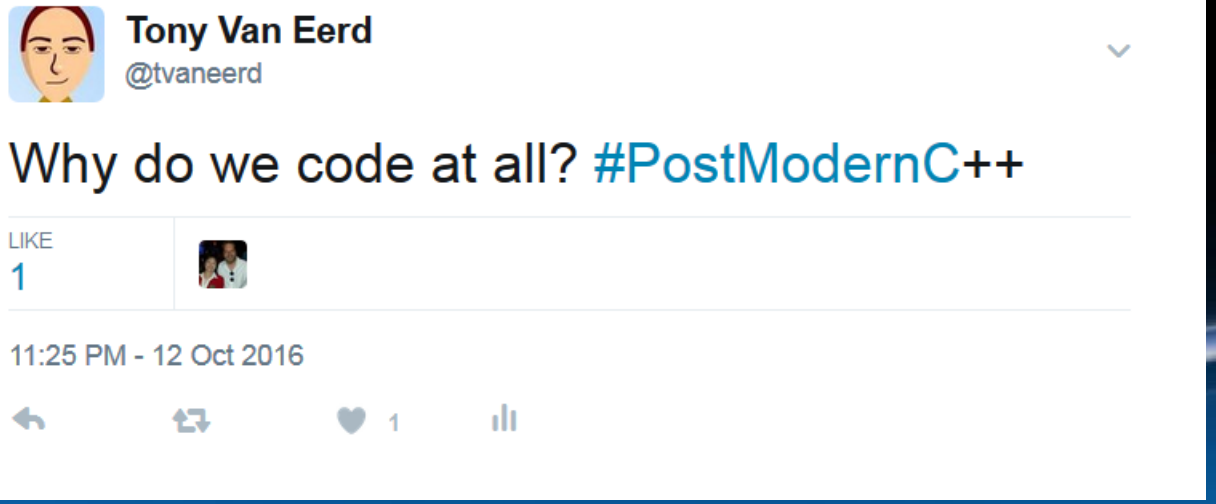
Join me tomorrow at #CppNow - I will be *taking meta-programming to a whole new level* cppnow2017.sched.com/event/A8Iq/pos… CLICKBAIT: It will SHOCK YOU.

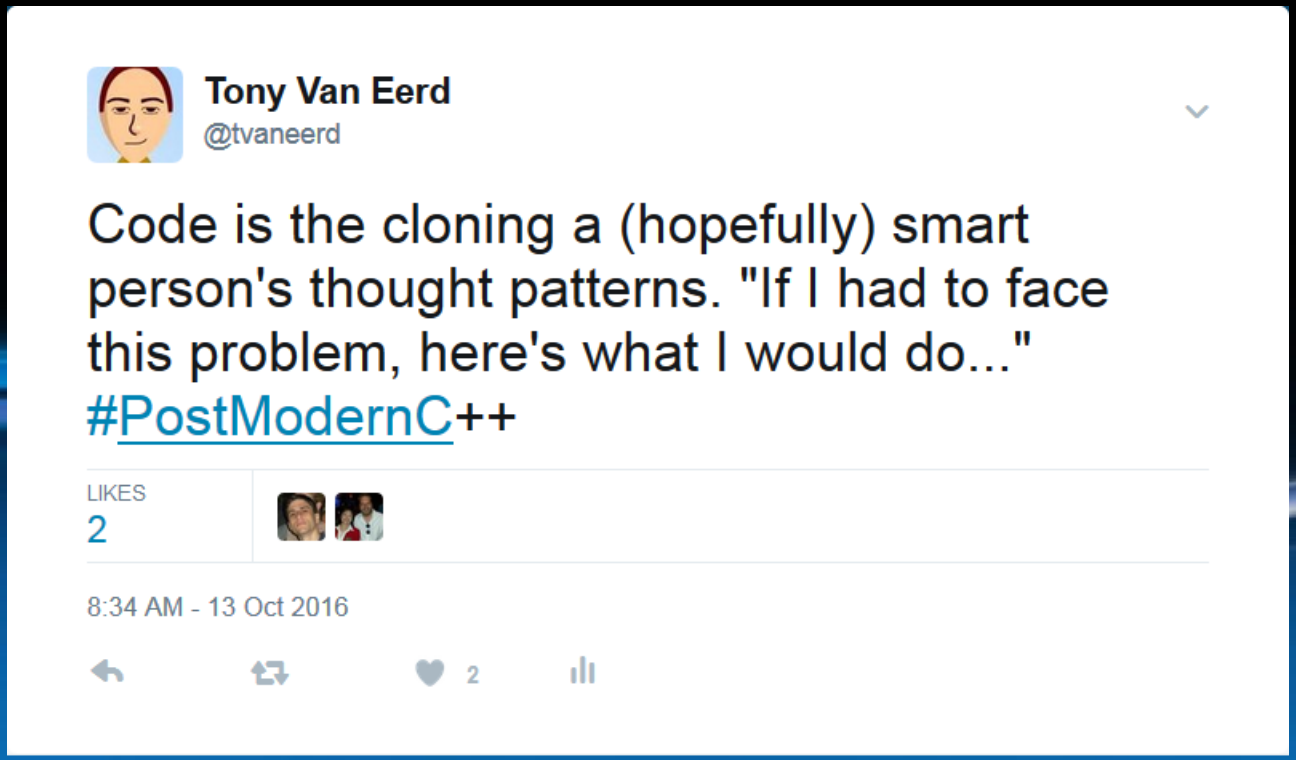
C++Now 2017: Postmodern C++
View more about this event at C++Now 2017
cppnow2017.sched.com

LIKE
1

12:38 PM - 17 May 2017

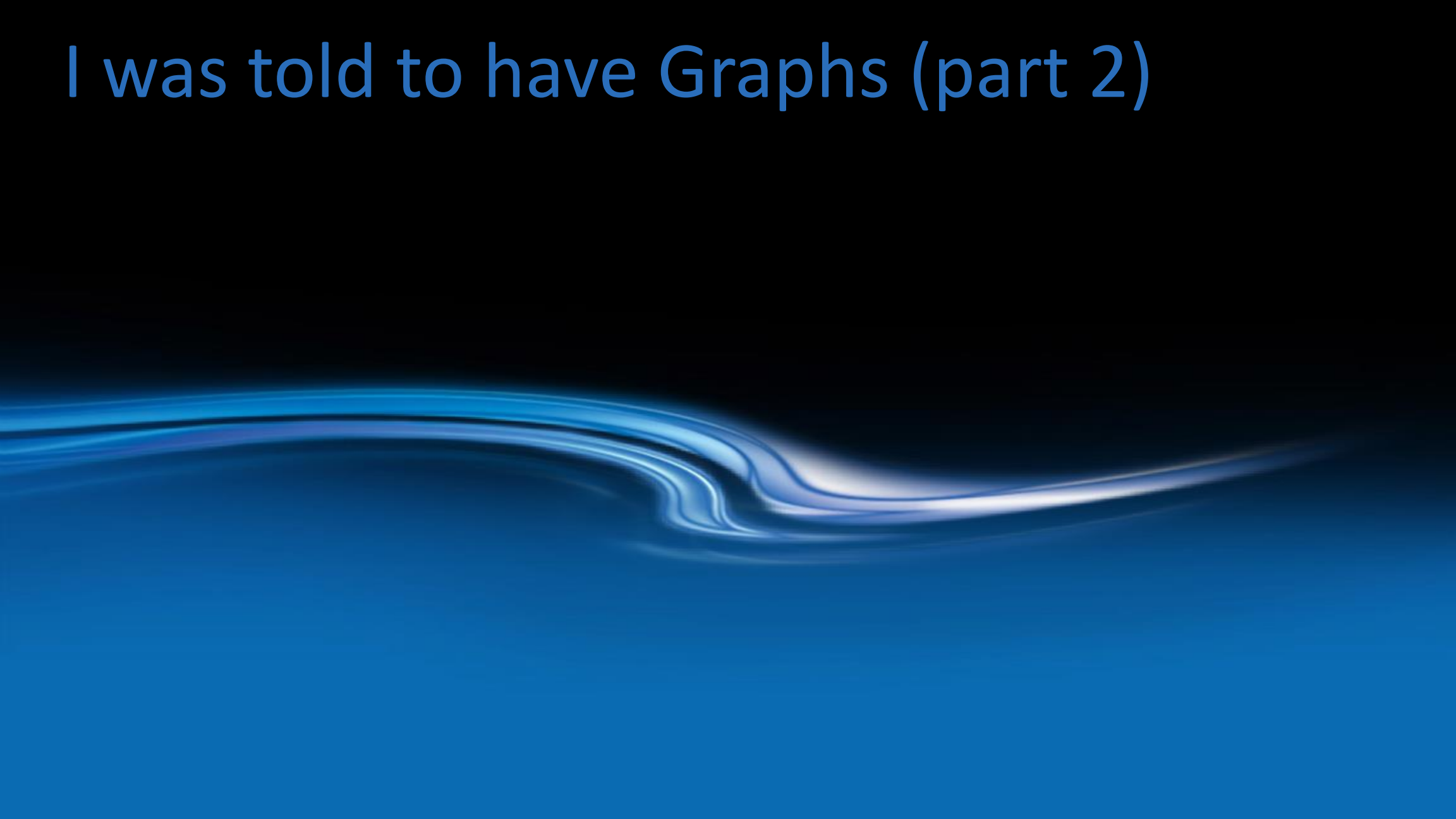


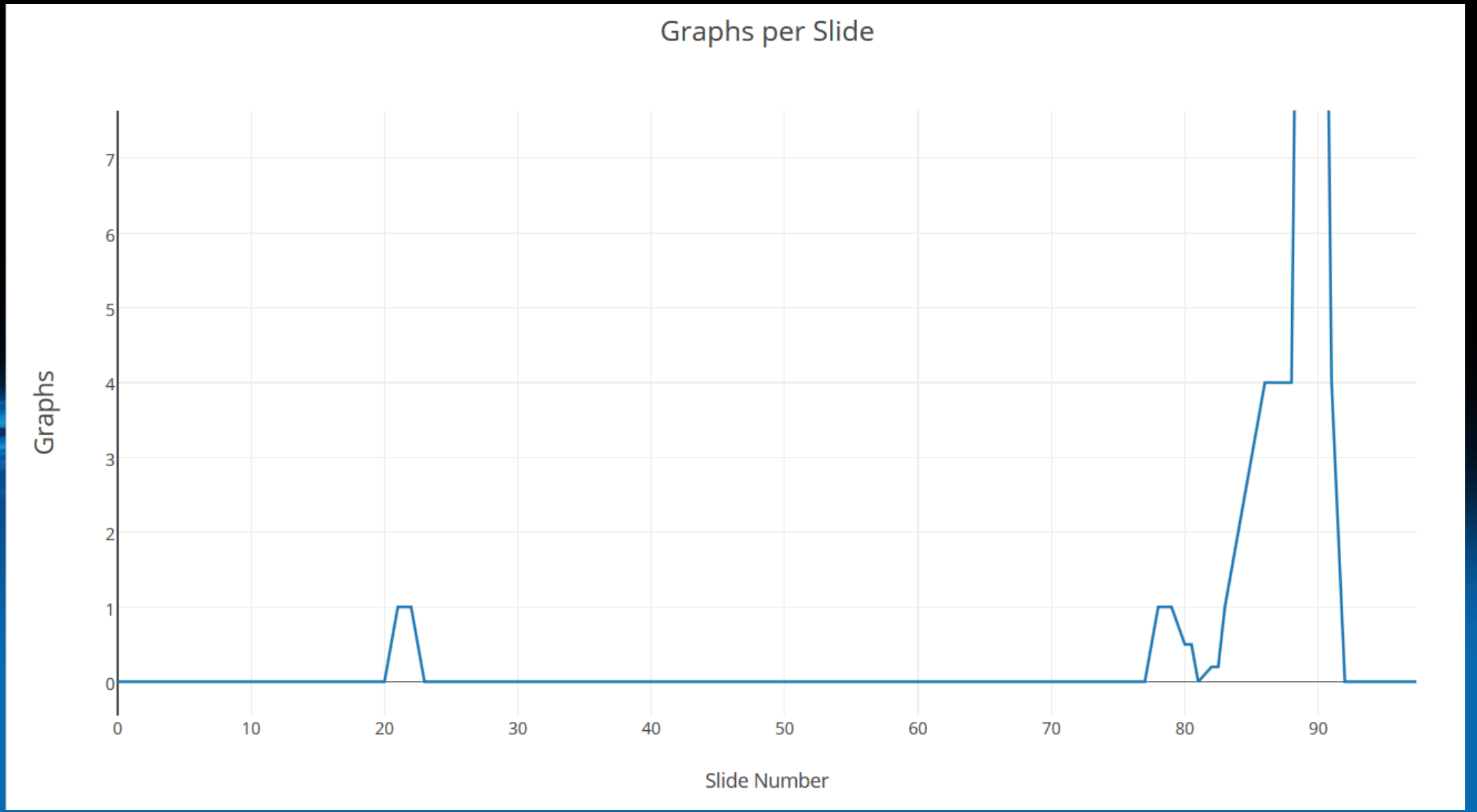

Tony Van Eerd
@tvaneerd
Why do we code at all? #PostModernC++
LIKE
1
11:25 PM - 12 Oct 2016
1

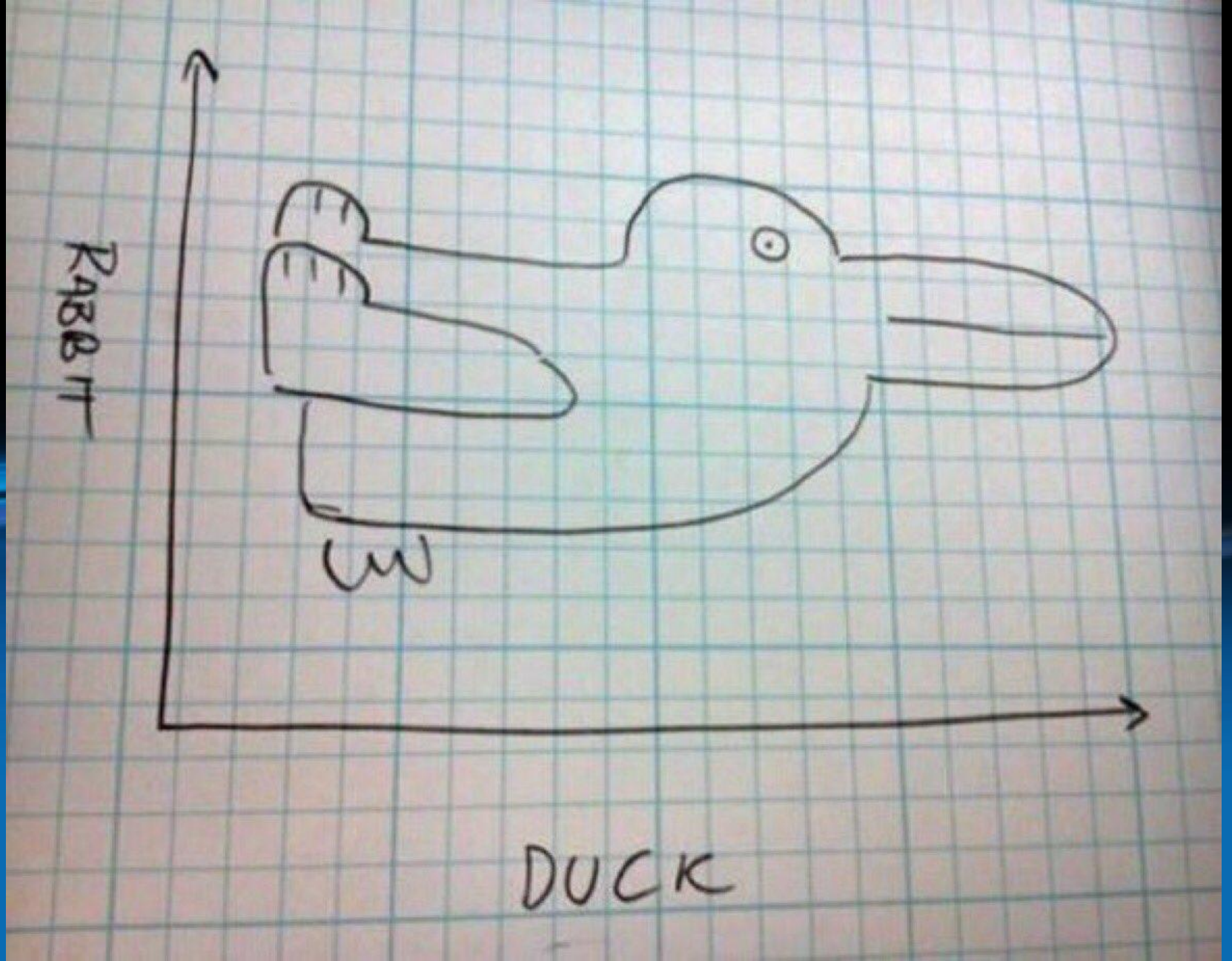


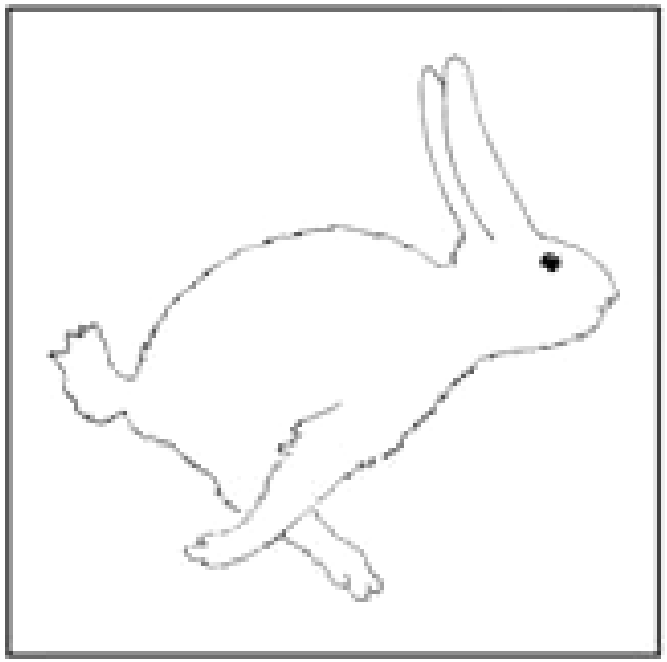
Adam Nevraumont

I was told to have Graphs (part 2)



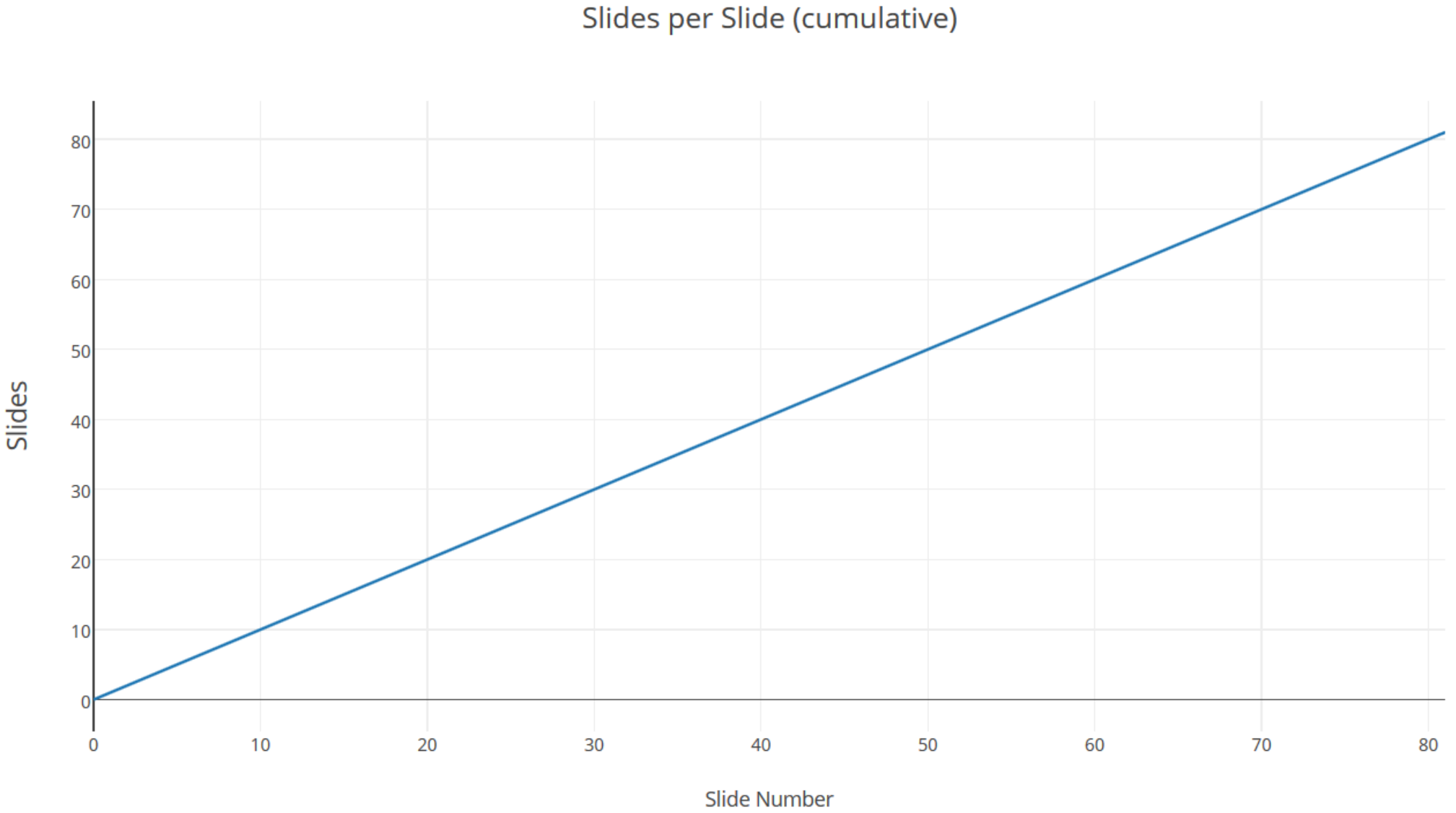

Graphs per Slide
Graphs
0
1
2
3
4
5
6
7
0
10
20
30
40
50
60
70
80
90
Slide Number

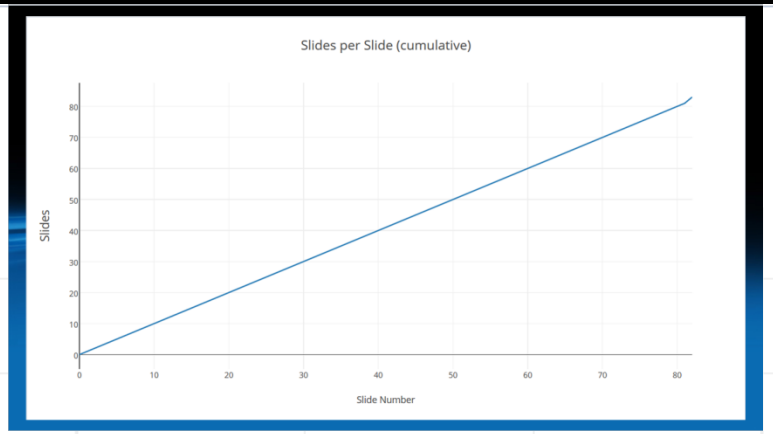

RABBIT
DUCK



23 October 1892 issue of Fliegende Blätter, a German humour magazine

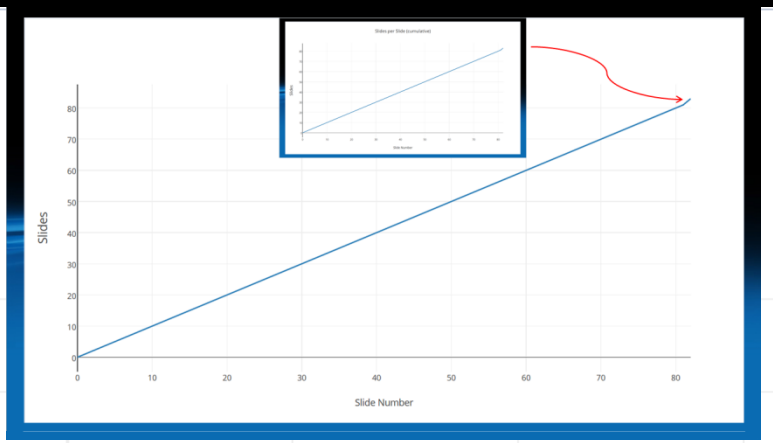
Photo by "Fir0002/Flagstaffotos"


Slides per Slide (cumulative)
Slides
0
10
20
30
40
50
60
70
80
Slide Number



Slides

Slide Number

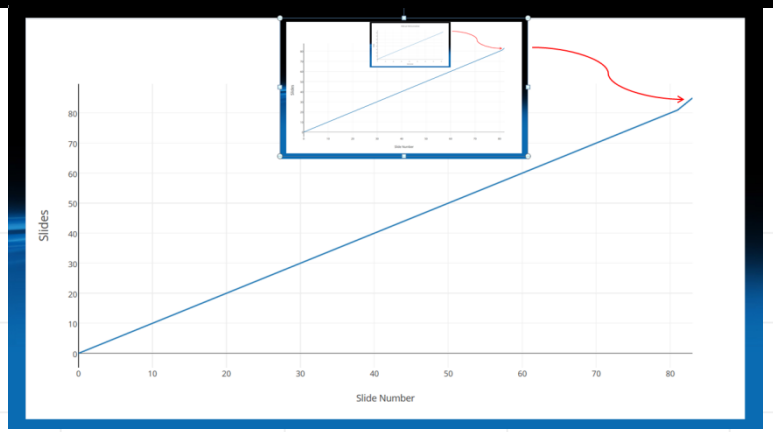


Slides

0
10
20
30
40
50
60
70
80

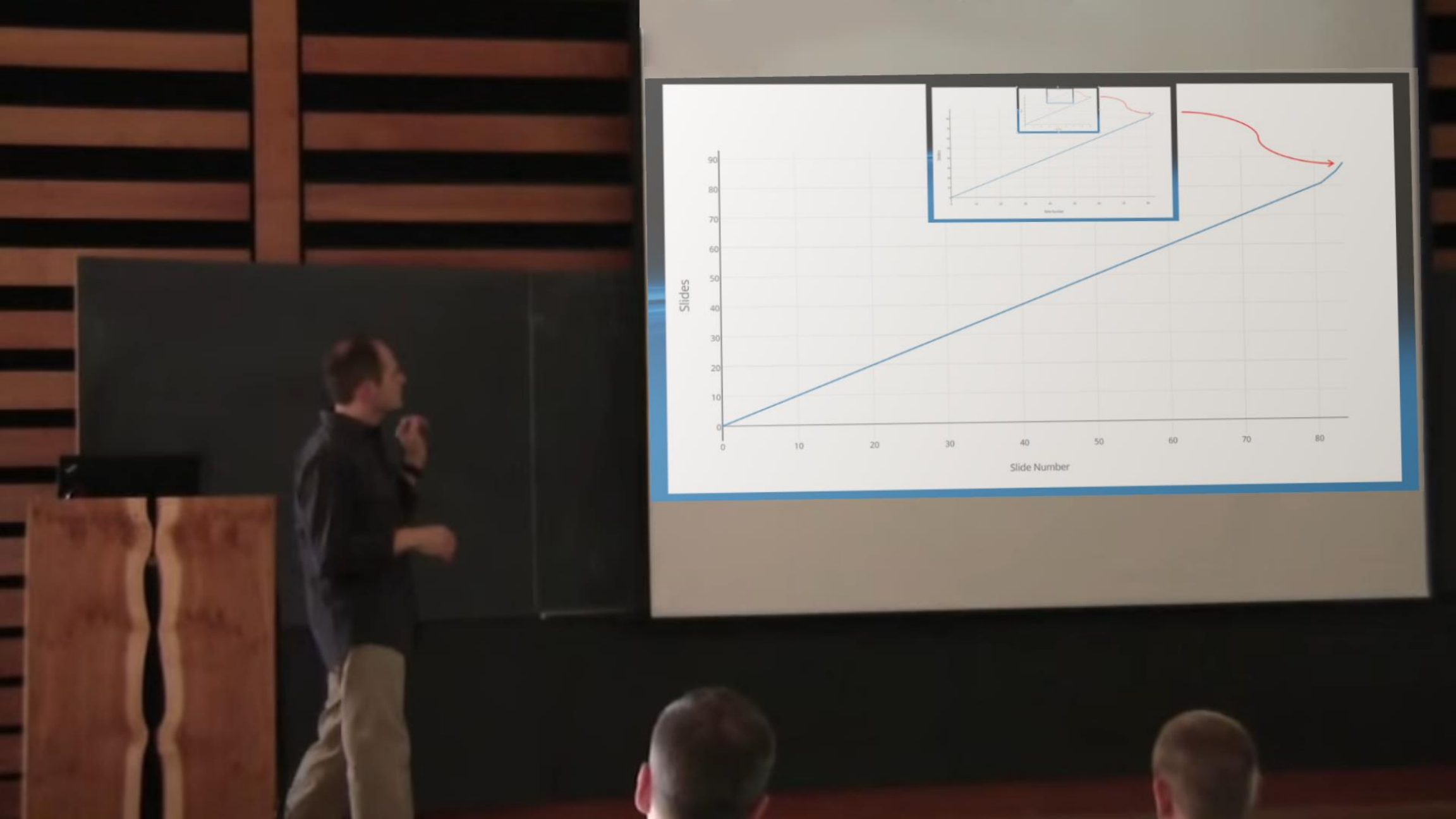
Slide Number

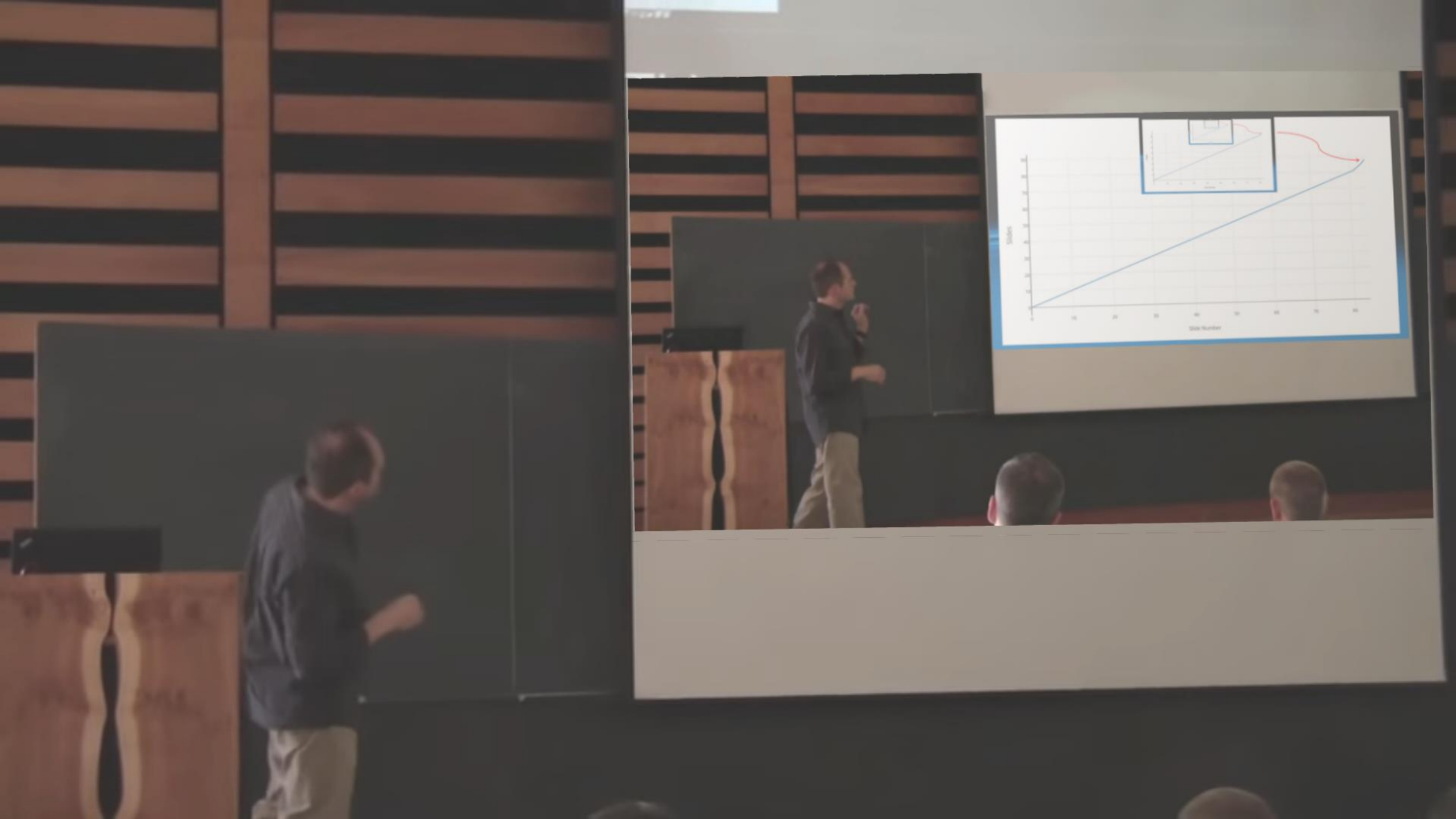
0
10
20
30
40
50
60
70
80



Slides

Slide Number


Slides
90
80
70
60
50
40
30
20
10
0
0
10
20
30
40
50
60
70
80
Slide Number

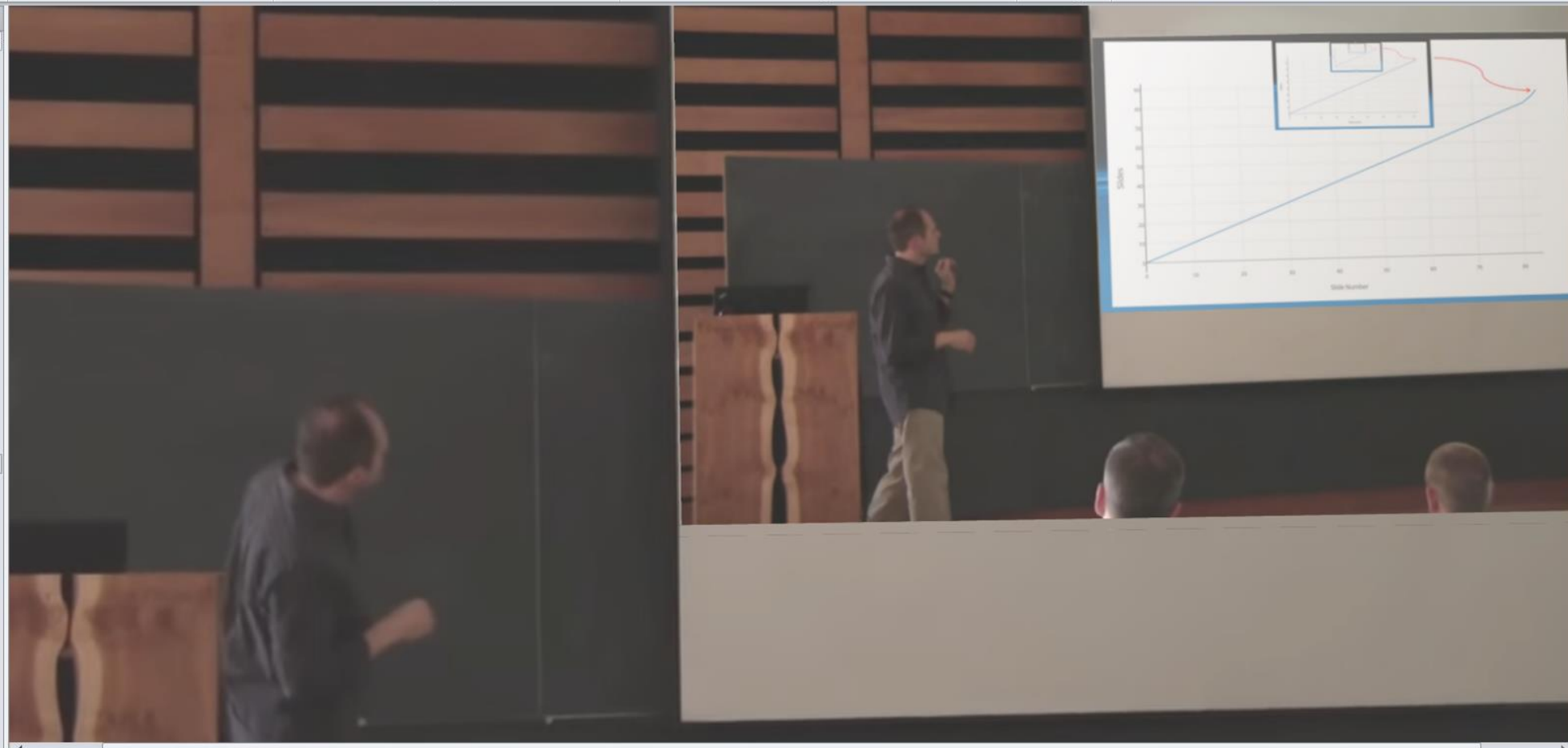

Slides
Slide Number

PostModernCpp.pptx - Microsoft PowerPoint

File Home Insert Design Transitions Animations Slide Show Review View Storyboarding

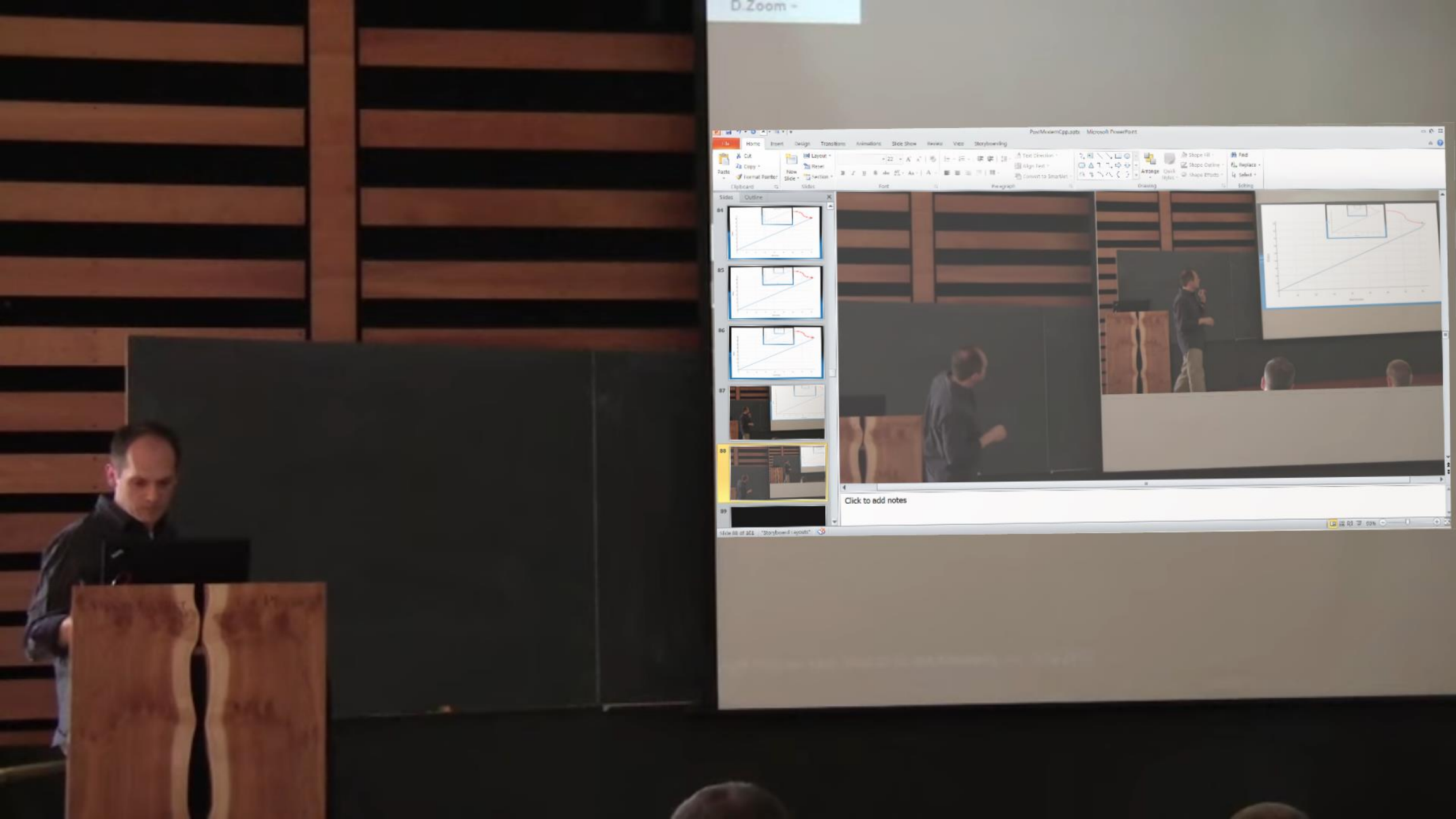
Cut Copy Format Painter — Clipboard
New Slide Layout Reset Section — Slides
Font
Paragraph
Text Direction Align Text Convert to SmartArt
Drawing
Arrange Quick Styles Shape Fill Shape Outline Shape Effects
Find Replace Select — Editing

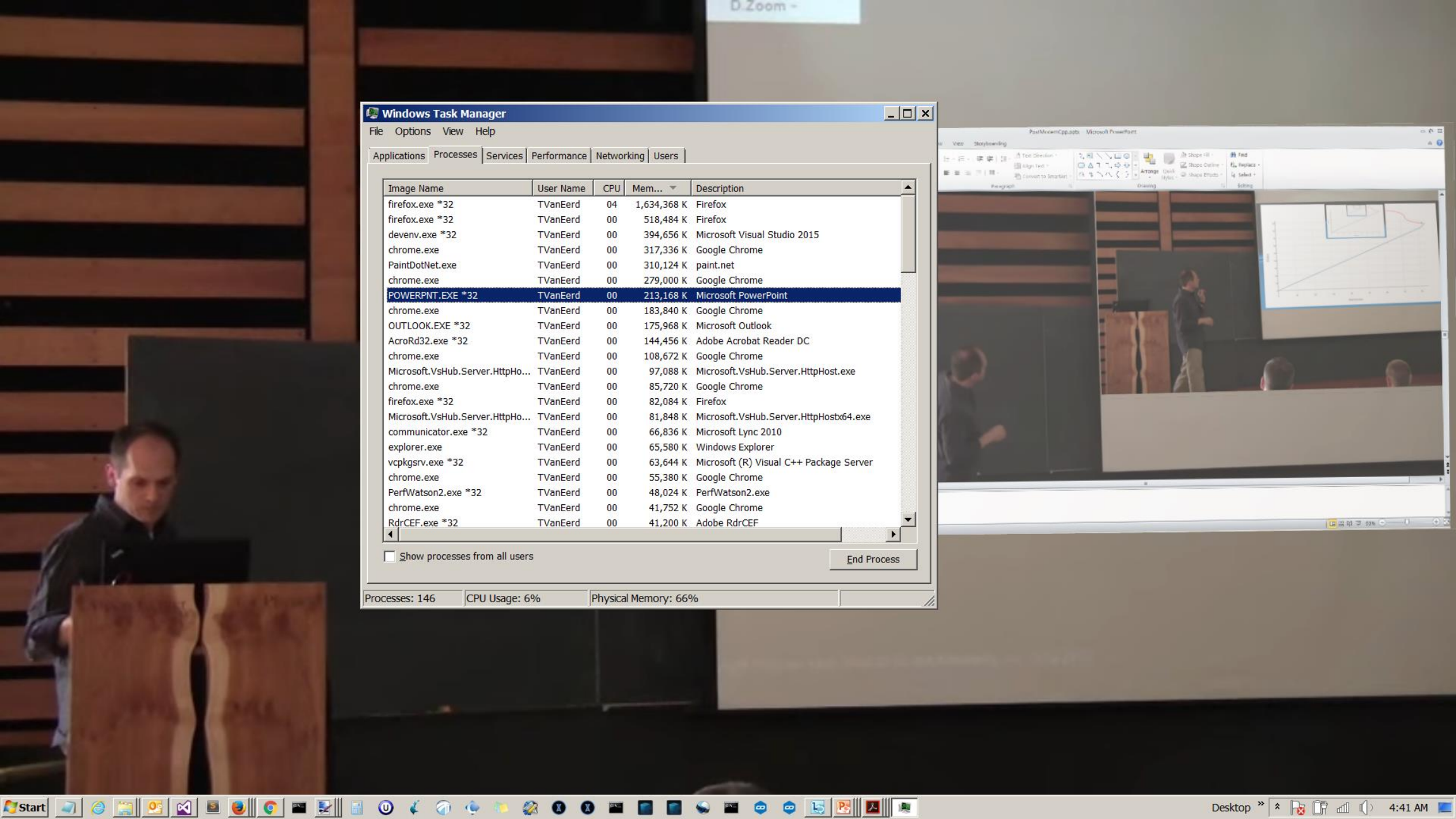
Slides Outline



Click to add notes

de 88 of 161 "Storyboard Layouts" 90%


D.Zoom -
Click to add notes


Windows Task Manager
File Options View Help
Applications Processes Services Performance Networking Users
Image Name | User Name | CPU | Mem... | Description
firefox.exe *32 | TVanEerd | 04 | 1,634,368 K | Firefox
firefox.exe *32 | TVanEerd | 00 | 518,484 K | Firefox
devenv.exe *32 | TVanEerd | 00 | 394,656 K | Microsoft Visual Studio 2015
chrome.exe | TVanEerd | 00 | 317,336 K | Google Chrome
PaintDotNet.exe | TVanEerd | 00 | 310,124 K | paint.net
chrome.exe | TVanEerd | 00 | 279,000 K | Google Chrome
POWERPNT.EXE *32 | TVanEerd | 00 | 213,168 K | Microsoft PowerPoint
chrome.exe | TVanEerd | 00 | 183,840 K | Google Chrome
OUTLOOK.EXE *32 | TVanEerd | 00 | 175,968 K | Microsoft Outlook
AcroRd32.exe *32 | TVanEerd | 00 | 144,456 K | Adobe Acrobat Reader DC
chrome.exe | TVanEerd | 00 | 108,672 K | Google Chrome
Microsoft.VsHub.Server.HttpHo... | TVanEerd | 00 | 97,088 K | Microsoft.VsHub.Server.HttpHost.exe
chrome.exe | TVanEerd | 00 | 85,720 K | Google Chrome
firefox.exe *32 | TVanEerd | 00 | 82,084 K | Firefox
Microsoft.VsHub.Server.HttpHo... | TVanEerd | 00 | 81,848 K | Microsoft.VsHub.Server.HttpHostx64.exe
communicator.exe *32 | TVanEerd | 00 | 66,836 K | Microsoft Lync 2010
explorer.exe | TVanEerd | 00 | 65,580 K | Windows Explorer
vcpkgsrv.exe *32 | TVanEerd | 00 | 63,644 K | Microsoft (R) Visual C++ Package Server
chrome.exe | TVanEerd | 00 | 55,380 K | Google Chrome
PerfWatson2.exe *32 | TVanEerd | 00 | 48,024 K | PerfWatson2.exe
chrome.exe | TVanEerd | 00 | 41,752 K | Google Chrome
RdrCEF.exe *32 | TVanEerd | 00 | 41,200 K | Adobe RdrCEF
Show processes from all users
End Process
Processes: 146
CPU Usage: 6%
Physical Memory: 66%
Start
Desktop
4:41 AM

Tony Van Eerd
@tvaneerd

Ponder:

```
class SelfReferential
{
  SelfReferential & selfReference;
public:
  SelfReferential() : selfReference(*this) { }
};
```

#PostModernCpp

RETWEETS	LIKES
5	16

7:24 AM - 18 Jan 2017

5 16

```
class SelfReferential
{
   SelfReferential & selfReference;
public:
   SelfReferential() :
      selfReference(*this) { }
};
```

```
class Self
{
   Self & self;
public:
   Self() :
      self(*this) { }
};
```

```
template <typename Derived> //CRTP
class Self
{
   Derived & self;
public:
   Self() :
      self(*this) { }
};
```

```
template <typename Derived>
class Self
{
   Derived & self;
public:
   Self() :
      self(*this) { }
};

class Foo : Self<Foo>
{
   int x;
   int getX() { return self.x; }
};
```

```
template <typename Derived>
class Self
{
protected:
    Derived & self;
public:
    Self() :
        self(*static_cast<Derived*>(this)) { }
};

class Foo : public Self<Foo>
{
    //friend Self<Foo>;
    int x;
public:
    int getX() { return self.x; }
};
```

“We Already Have Enough Paradigms”

Paul (Pablo) Young

“We Already Have Enough Paradigms”

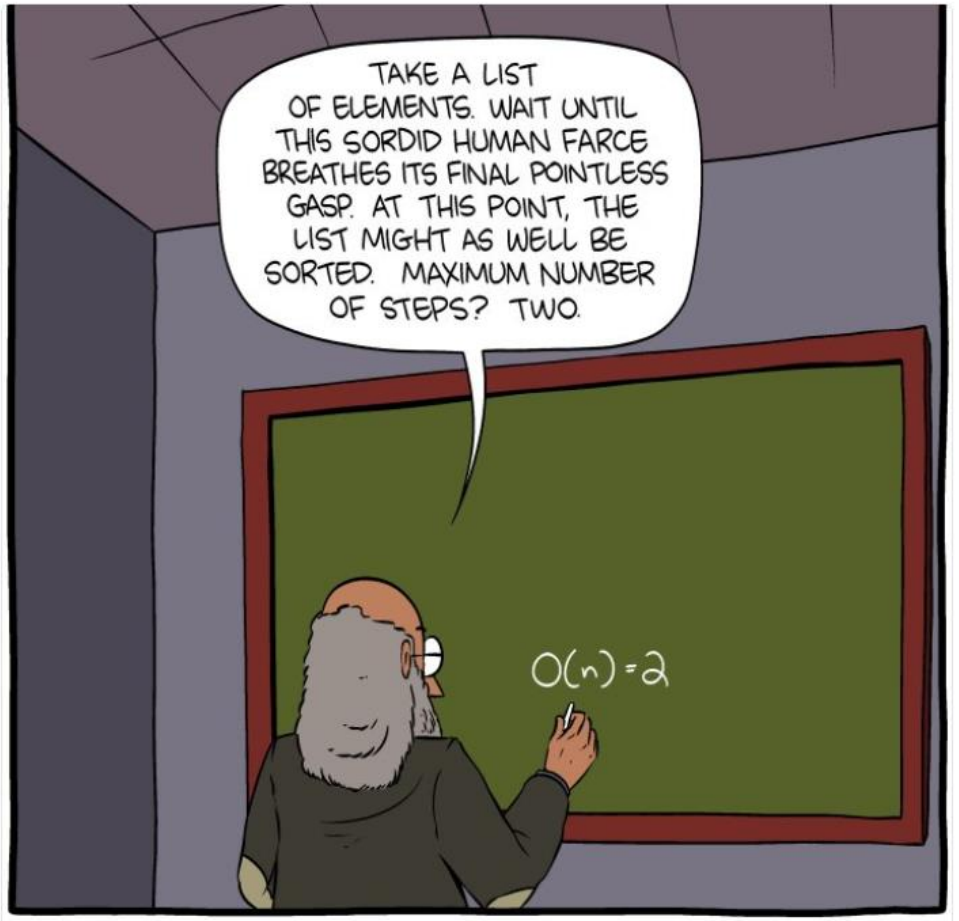
Paul (Pablo) Young

```
{
    …
    new Widget(x,y,w,h, parent);
    new Button(10,10,50,20, “OK”, parent);
    …
}
```

Zach Weinersmith ✓
@ZachWeiner

Follow

Efficient Sorting
smbc-comics.com/comic/efficien …
#smbc #hiveworks



Tony Van Eerd
@tvaneerd

Replying to @ZachWeiner @rodgertq

Every list is already sorted - from a certain point of view. Number steps = 0. (except finding the POV) #PostModernCpp

LIKE
1

4:15 PM - 17 Mar 2017

1

?

```
sort(v.begin(), v.end(),
   [](auto const & a, auto const &b)
   {
      return std::less(&a, &b);
   });
```

Tony Van Eerd
@tvaneerd

How to "concentrate on one section of a programme at a time" without intrusion, and how to "test in isolation" here:

C++Now 2017: Postmodern C++
View more about this event at C++Now 2017
cppnow2017.sched.com

3:50 PM - 17 May 2017

Tony Van Eerd
@tvaneerd

Postmodernism began in 1951, when David Wheeler (@stroustrup's thesis adviser) invented the subroutine. I hope it catches on. #PostModernCpp

8:48 PM - 6 Mar 2017

THE USE OF SUB-ROUTINES IN PROGRAMMES

D. J. Wheeler

Cambridge & Illinois Universities

A sub-routine may perhaps best be described as a self-contained part of a programme, which is capable of being used in different programmes. It is an entity of its own within a programme. There is no necessity to compose a programme of a set of distinct sub-routines; for the programme can be written as a complete unit, with no divisions into smaller parts. However it is usually advantageous to arrange that a programme is comprised of a set of sub-routines, some of which have been made specially for the particular programme while others are available from a 'library' of standard sub-routines. The reasons for this will be discussed below.

When a programme has been made from a set of sub-routines the breakdown of the code is more complete than it would otherwise be. This allows the coder to concentrate on one section of a programme at a time without the overall detailed programme continually intruding. Thus the sub-routines can be more easily coded and the tested in isolation from the rest of the programme. When the entire programme has to be tested it is with the foreknowledge that the incidence of mistakes in the sub-routines is zero (or at least one order of magnitude below that of the untested portions of the programme!)

If library sub-routines exist for the major part of a code then the task of constructing the easier to use a sub-routine which will meet the specifications with a small amount of manipulation than to make one specially for the purpose.

It should be pointed out that the preparation of a library sub-routine requires a considerable amount of work. This is much greater than the effort merely required to code the sub-routine in its simplest possible form. It will usually be necessary to code it in the library standard form and this may detract from its efficiency in time and space. It may be desirable to code it in such a manner that the operation is generalized to some extent. However, even after it has been coded and tested there still remains the considerable task of writing a description so that people not acquainted with the interior coding can nevertheless use it easily. This last task may be the most difficult.

Besides the organization of the individual sub-routines there remains the method of the general organization of the library. How are the sub-routines going to be stored? Are they going to be stored on punched paper tape or are they going to be available in the auxiliary store of the machine? Usually it will be found that it is not possible to write the sub-routines such that they may be put into arbitrary positions in the store-although in certain machines this is now possible. Usually some translation process will have to be

routines can be more easily coded and the tested in isolation from the rest of the programme. When the entire programme has to be tested it is with the foreknowledge that the incidence of mistakes in the sub-routines is zero (or at least one order of magnitude below that of the untested portions of the programme:)

If library sub-routines exist for the major part of a code then the task of constructing the remaining part of the programme is naturally very much less than if the code had to be written from the very beginning. However, one will rarely have available sub-routines to do exactly what is required and thus a certain amount of manipulation may be necessary before a given sub-routine can be used. Even so,it is usually far

organization of the library. How are the sub-routines going to be stored? Are they going to be stored on punched paper tape or are they going to be available in the auxiliary store of the machine? Usually it will be found that it is not possible to write the sub-routines such that they may be put into arbitrary positions in the store-although in certain machines this is now possible. Usually some translation process will have to be arranged so that an invariant form of sub-routine stored on some medium such as paper tape can be translated to the form required in a particular application. This translation is possible because fixed rules can be set up for adjusting a sub-routine so that it becomes correct in the set of locations in which it is put and used.

One next considers the methods by which sub-routines can be used. There are a number of different ways of transferring control to sub-routines and arranging that control is returned to the appropriate point to which it is required. One of the simpler methods was that used for the closed sub-routines of the EDSAC in which it was arranged that when the sub-routine had performed its part of the computation then control was returned to a point in the main programme immediately after the orders which had called it into use. This has been described in detail by Goldstine. This perhaps facilitates thinking of a sub-

'orders' that are obeyed are identical with those of the machine. However, the interpretive routine retains control and so it is possible to print out extra information about the course of the programme. This extra information makes it possible to follow the meanderings of the program in detail thus helping to locate the errors of a programme. This is not a good method of finding errors in programmes as it takes a long time and the programmers knowledge of the programme is not utilized - as it should be - in tracing the fault. However, it is a useful last resort and can quite often give out information about a code which would be difficult to find

EDSAC in which it was arranged that when the sub-routine had performed its part of the computation then control was returned to a point in the main programme immediately after the orders which had called it into use. This has been described in detail by Goldstine. This perhaps facilitates thinking of a sub-routine as an 'order' of the machine although it is usually of a more complicated kind than that wired in the circuits of the machine.

A second more interesting type of sub-routine is an interpretive routine. In this type of routine it is arranged that a sequence of operations is performed each time the sub-routine is called into action, each operation being determined by one parameter or 'order' in a list of such 'orders'. This type of sub-routine is particularly useful for coding certain special types of arithmetic for the machine, for example, floating point arithmetic in which numbers are expressed as $\mathbf{X} \times 10^{p}$. Thus the sub-routine executes the 'orders' in the list in a similar fashion to the way that the machine obeys ordinary orders. However, the orders that it does are determined by the parts of the sub-routine, and so can be made to do any kind of operation or arithmetic.

One extension of an interpretive routine is a checking routine which is so arranged that the

of a programme. This is not a good method of finding errors in programmes as it takes a long time and the programmers knowledge of the programme is not utilized - as it should be - in tracing the fault. However, it is a useful last resort and can quite often give out information about a code which would be difficult to find in any other way.

Sub-routines seem to have two distinct uses in programmes. The first and most obvious use is for the evaluation of functions, a simple example being the evaluation of sine x given x. The second use is for the organization of processes such as the integration of a function given f(x). This second type requires more consideration to make it useful and general. For instance how should f(x) be specified for the sub-routine? One obvious and useful way is to allow the integrating sub-routine access to an auxiliary sub-routine which is capable of evaluating f(x).

The above remarks may be summarized by saying sub-routines are very useful-although not absolutely necessary-and that the prime objectives to be born in mind when constructing them are simplicity of use, correctness of codes and accuracy of description. All complexities should-if possible -be buried out of sight.

"self-contained part of a programme"

"an entity of its own"

THE USE OF SUB-ROUTINES IN PROGRAMMES

D. J. Wheeler

Cambridge & Illinois Universities

A sub-routine may perhaps best be described as a self-contained part of a programme, which is capable of being used in different programmes. It is an entity of its own within a programme. There is no necessity to compose a programme of a set of distinct sub-routines; for the programme can be written as a complete unit, with no divisions into smaller parts. However it is usually advantageous to arrange that a programme is comprised of a set of sub-routines, some of which have been made specially for the particular programme while others are available from a 'library' of standard sub-routines. The reasons for this will be discussed below.

When a programme has been made from a set of sub-routines the breakdown of the code is more complete than it would otherwise be. This allows the coder to concentrate on one section of a programme at a time without the overall detailed programme continually intruding. Thus the sub-routines can be more easily coded and the tested in isolation from the rest of the programme. When the entire programme has to be tested it is with the foreknowledge that the incidence of mistakes in the sub-routines is zero (or at least one order of magnitude below that of the untested portions of the programme!)

If library sub-routines exist for the major part of a code then the task of constructing the easier to use a sub-routine which will meet the specifications with a small amount of manipulation than to make one specially for the purpose.

It should be pointed out that the preparation of a library sub-routine requires a considerable amount of work. This is much greater than the effort merely required to code the sub-routine in its simplest possible form. It will usually be necessary to code it in the library standard form and this may detract from its efficiency in time and space. It may be desirable to code it in such a manner that the operation is generalized to some extent. However, even after it has been coded and tested there still remains the considerable task of writing a description so that people not acquainted with the interior coding can nevertheless use it easily. This last task may be the most difficult.

Besides the organization of the individual sub-routines there remains the method of the general organization of the library. How are the sub-routines going to be stored? Are they going to be stored on punched paper tape or are they going to be available in the auxiliary store of the machine? Usually it will be found that it is not possible to write the sub-routines such that they may be put into arbitrary positions in the store-although in certain machines this is now possible. Usually some translation process will have to be

"There is no necessity…" to use sub-routines

…

"However it is usually advantangeous"

"The reason for this will be discussed below"

THE USE OF SUB-ROUTINES IN PROGRAMMES

D. J. Wheeler

Cambridge & Illinois Universities

A sub-routine may perhaps best be described as a self-contained part of a programme, which is capable of being used in different programmes. It is an entity of its own within a programme. There is no necessity to compose a programme of a set of distinct sub-routines; for the programme can be written as a complete unit, with no divisions into smaller parts. However it is usually advantageous to arrange that a programme is comprised of a set of sub-routines, some of which have been made specially for the particular programme while others are available from a 'library' of standard sub-routines. The reasons for this will be discussed below.

When a programme has been made from a set of sub-routines the breakdown of the code is more complete than it would otherwise be. This allows the coder to concentrate on one section of a programme at a time without the overall detailed programme continually intruding. Thus the sub-routines can be more easily coded and the tested in isolation from the rest of the programme. When the entire programme has to be tested it is with the foreknowledge that the incidence of mistakes in the sub-routines is zero (or at least one order of magnitude below that of the untested portions of the programme!)

If library sub-routines exist for the major part of a code then the task of constructing the easier to use a sub-routine which will meet the specifications with a small amount of manipulation than to make one specially for the purpose.

It should be pointed out that the preparation of a library sub-routine requires a considerable amount of work. This is much greater than the effort merely required to code the sub-routine in its simplest possible form. It will usually be necessary to code it in the library standard form and this may detract from its efficiency in time and space. It may be desirable to code it in such a manner that the operation is generalized to some extent. However, even after it has been coded and tested there still remains the considerable task of writing a description so that people not acquainted with the interior coding can nevertheless use it easily. This last task may be the most difficult.

Besides the organization of the individual sub-routines there remains the method of the general organization of the library. How are the sub-routines going to be stored? Are they going to be stored on punched paper tape or are they going to be available in the auxiliary store of the machine? Usually it will be found that it is not possible to write the sub-routines such that they may be put into arbitrary positions in the store-although in certain machines this is now possible. Usually some translation process will have to be

"allows the coder to concentrate on one section of a programme at a time without the overall detailed programme continually intruding"

THE USE OF SUB-ROUTINES IN PROGRAMMES

D. J. Wheeler

Cambridge & Illinois Universities

A sub-routine may perhaps best be described as a self-contained part of a programme, which is capable of being used in different programmes. It is an entity of its own within a programme. There is no necessity to compose a programme of a set of distinct sub-routines; for the programme can be written as a complete unit, with no divisions into smaller parts. However it is usually advantageous to arrange that a programme is comprised of a set of sub-routines, some of which have been made specially for the particular programme while others are available from a 'library' of standard sub-routines. The reasons for this will be discussed below.

When a programme has been made from a set of sub-routines the breakdown of the code is more complete than it would otherwise be. This allows the coder to concentrate on one section of a programme at a time without the overall detailed programme continually intruding. Thus the sub-routines can be more easily coded and the tested in isolation from the rest of the programme. When the entire programme has to be tested it is with the foreknowledge that the incidence of mistakes in the sub-routines is zero (or at least one order of magnitude below that of the untested portions of the programme!)

If library sub-routines exist for the major part of a code then the task of constructing the easier to use a sub-routine which will meet the specifications with a small amount of manipulation than to make one specially for the purpose.

It should be pointed out that the preparation of a library sub-routine requires a considerable amount of work. This is much greater than the effort merely required to code the sub-routine in its simplest possible form. It will usually be necessary to code it in the library standard form and this may detract from its efficiency in time and space. It may be desirable to code it in such a manner that the operation is generalized to some extent. However, even after it has been coded and tested there still remains the considerable task of writing a description so that people not acquainted with the interior coding can nevertheless use it easily. This last task may be the most difficult.

Besides the organization of the individual sub-routines there remains the method of the general organization of the library. How are the sub-routines going to be stored? Are they going to be stored on punched paper tape or are they going to be available in the auxiliary store of the machine? Usually it will be found that it is not possible to write the sub-routines such that they may be put into arbitrary positions in the store-although in certain machines this is now possible. Usually some translation process will have to be

"Thus the subroutines can be more **easily coded and tested in isolation** from the rest of the programme."

THE USE OF SUB-ROUTINES IN PROGRAMMES

D. J. Wheeler

Cambridge & Illinois Universities

A sub-routine may perhaps best be described as a self-contained part of a programme, which is capable of being used in different programmes. It is an entity of its own within a programme. There is no necessity to compose a programme of a set of distinct sub-routines; for the programme can be written as a complete unit, with no divisions into smaller parts. However it is usually advantageous to arrange that a programme is comprised of a set of sub-routines, some of which have been made specially for the particular programme while others are available from a 'library' of standard sub-routines. The reasons for this will be discussed below.

When a programme has been made from a set of sub-routines the breakdown of the code is more complete than it would otherwise be. This allows the coder to concentrate on one section of a programme at a time without the overall detailed programme continually intruding. Thus the sub-routines can be more easily coded and the tested in isolation from the rest of the programme. When the entire programme has to be tested it is with the foreknowledge that the incidence of mistakes in the sub-routines is zero (or at least one order of magnitude below that of the untested portions of the programme!)

If library sub-routines exist for the major part of a code then the task of constructing the easier to use a sub-routine which will meet the specifications with a small amount of manipulation than to make one specially for the purpose.

It should be pointed out that the preparation of a library sub-routine requires a considerable amount of work. This is much greater than the effort merely required to code the sub-routine in its simplest possible form. It will usually be necessary to code it in the library standard form and this may detract from its efficiency in time and space. It may be desirable to code it in such a manner that the operation is generalized to some extent. However, even after it has been coded and tested there still remains the considerable task of writing a description so that people not acquainted with the interior coding can nevertheless use it easily. This last task may be the most difficult.

Besides the organization of the individual sub-routines there remains the method of the general organization of the library. How are the sub-routines going to be stored? Are they going to be stored on punched paper tape or are they going to be available in the auxiliary store of the machine? Usually it will be found that it is not possible to write the sub-routines such that they may be put into arbitrary positions in the store-although in certain machines this is now possible. Usually some translation process will have to be

LIBRARIES!!!
(a bunch of sub-routines in some kind of package separate from any one programme, **which can be reused from one programme to the next!!!)**

THE USE OF SUB-ROUTINES IN PROGRAMMES

D. J. Wheeler
Cambridge & Illinois Universities

A sub-routine may perhaps best be described as a self-contained part of a programme, which is capable of being used in different programmes. It is an entity of its own within a programme. There is no necessity to compose a programme of a set of distinct sub-routines; for the programme can be written as a complete unit, with no divisions into smaller parts. However it is usually advantageous to arrange that a programme is comprised of a set of sub-routines, some of which have been made specially for the particular programme while others are available from a 'library' of standard sub-routines. The reasons for this will be discussed below.

When a programme has been made from a set of sub-routines the breakdown of the code is more complete than it would otherwise be. This allows the coder to concentrate on one section of a programme at a time without the overall detailed programme continually intruding. Thus the sub-routines can be more easily coded and the tested in isolation from the rest of the programme. When the entire programme has to be tested it is with the foreknowledge that the incidence of mistakes in the sub-routines is zero (or at least one order of magnitude below that of the untested portions of the programme!)

If library sub-routines exist for the major part of a code then the task of constructing the easier to use a sub-routine which will meet the specifications with a small amount of manipulation than to make one specially for the purpose.

It should be pointed out that the preparation of a library sub-routine requires a considerable amount of work. This is much greater than the effort merely required to code the sub-routine in its simplest possible form. It will usually be necessary to code it in the library standard form and this may detract from its efficiency in time and space. It may be desirable to code it in such a manner that the operation is generalized to some extent. However, even after it has been coded and tested there still remains the considerable task of writing a description so that people not acquainted with the interior coding can nevertheless use it easily. This last task may be the most difficult.

Besides the organization of the individual sub-routines there remains the method of the general organization of the library. How are the sub-routines going to be stored? Are they going to be stored on punched paper tape or are they going to be available in the auxiliary store of the machine? Usually it will be found that it is not possible to write the sub-routines such that they may be put into arbitrary positions in the store-although in certain machines this is now possible. Usually some translation process will have to be

THE USE OF SUB-ROUTINES IN PROGRAMMES

D. J. Wheeler

Cambridge & Illinois Universities

A sub-routine may perhaps best be described as a self-contained part of a programme, which is capable of being used in different programmes. It is an entity of its own within a programme. There is no necessity to compose a programme of a set of distinct sub-routines; for the programme can be written as a complete unit, with no divisions into smaller parts. However it is usually advantageous to arrange that a programme is comprised of a set of sub-routines, some of which have been made specially for the particular programme while others are available from a 'library' of standard sub-routines. The reasons for this will be discussed below.

When a programme has been made from a set of sub-routines the breakdown of the code is more complete than it would otherwise be. This allows the coder to concentrate on one section of a programme at a time without the overall detailed programme continually intruding. Thus the sub-routines can be more easily coded and the tested in isolation from the rest of the programme. When the entire programme has to be tested it is with the foreknowledge that the incidence of mistakes in the sub-routines is zero (or at least one order of magnitude below that of the untested portions of the programme!)

If library sub-routines exist for the major part of a code then the task of constructing the easier to use a sub-routine which will meet the specifications with a small amount of manipulation than to make one specially for the purpose.

It should be pointed out that the preparation of a library sub-routine requires a considerable amount of work. This is much greater than the effort merely required to code the sub-routine in its simplest possible form. It will usually be necessary to code it in the library standard form and this may detract from its efficiency in time and space. It may be desirable to code it in such a manner that the operation is generalized to some extent. However, even after it has been coded and tested there still remains the considerable task of writing a description so that people not acquainted with the interior coding can nevertheless use it easily. This last task may be the most difficult.

Besides the organization of the individual sub-routines there remains the method of the general organization of the library. How are the sub-routines going to be stored? Are they going to be stored on punched paper tape or are they going to be available in the auxiliary store of the machine? Usually it will be found that it is not possible to write the sub-routines such that they may be put into arbitrary positions in the store-although in certain machines this is now possible. Usually some translation process will have to be

Usually it will be found that it is not possible to write the sub-routines such that they may be put into arbitrary positions in the store – although in certain machines this is now possible.

SEPARATION OF CONCERNS

Find all "step 1", Subfolders, Find Results 1, Entire Solution, ""

...

...

...

...

Matching lines: **20** Matching files: 12 Total files searched: **3202**

```
...
...
// step 1, turn 2D camera pts into 3D points
std::vector<Point3D> cam3d[2];
std::vector<Point2D> proj2d[2];
auto& camData = getCamData();
auto& projData = getProjData();
for (int ptIdx = 0; ptIdx < camData.size(); ptIdx++)
{
    auto& camPoint = camData[ptIdx];
    Point3D p1;
    Point3D p2;
    m_camera->mapPointTo3D(camPoint, p1, p2, calibrationType);
    auto iset = sg->intersect(p1, p2);

    for (auto& intersection : iset)
    {
       ...
       ...
    }
}
...
...
// step 2, solve via mapper
...
// step 3, ...
...
// validate fit
...
```

Find all "step 1", Subfolders, Find Results 1, Entire Solution, ""
...
...
...
...
Matching lines: **20** Matching files: 12 Total files searched: **3202**

Between 3 and 40 lines per step

```
// step 1
{
   ...
   ...
   ...
   ...
   ...
}
```

```
// Step 1.      Step 2.      Step 3.
// ##oo         #o#o         ####
// ##oo  -->    oooo   -->   ####
// oooo         #o#o         ####
// oooo         oooo         ####
```

Find all "step 1", Subfolders, Find Results 1, Entire Solution, ""

...

...

C:\Code\AutoCal_trunk\Framework\CXX\Transforms\ControlCurve.cpp(1085):// Step 1: Mirror all the points

C:\Code\AutoCal_trunk\Framework\CXX\Transforms\ControlCurve.cpp(1112):// Step 1: Mirror all the points

C:\Code\AutoCal_trunk\Framework\CXX\Transforms\CubicControlCurve.cpp(223):// Step 1: Mirror all the points

C:\Code\AutoCal_trunk\Framework\CXX\Transforms\CubicControlCurve.cpp(262):// Step 1: Mirror all the points

C:\Code\AutoCal_trunk\Framework\CXX\Transforms\mesh.cpp(656):// Step 1: Mirror all the points

C:\Code\AutoCal_trunk\Framework\CXX\Transforms\mesh.cpp(712):// Step 1: Mirror all the points

...

...

Matching lines: **20** Matching files: 12 Total files searched: **3202**

```
Find all "step 1", Subfolders, Find Results 1, Entire Solution, ""
...
...
C:\Code\AutoCal_trunk\Framework\CXX\Transforms\ControlCurve.cpp(1085):// Step 1: Mirror all the points
C:\Code\AutoCal_trunk\Framework\CXX\Transforms\ControlCurve.cpp(1112):// Step 1: Mirror all the points
C:\Code\AutoCal_trunk\Framework\CXX\Transforms\CubicControlCurve.cpp(223):// Step 1: Mirror all the points
C:\Code\AutoCal_trunk\Framework\CXX\Transforms\CubicControlCurve.cpp(262):// Step 1: Mirror all the points
C:\Code\AutoCal_trunk\Framework\CXX\Transforms\mesh.cpp(656):// Step 1: Mirror all the points
C:\Code\AutoCal_trunk\Framework\CXX\Transforms\mesh.cpp(712):// Step 1: Mirror all the points
...
...
Matching lines: 20    Matching files: 12    Total files searched: 3202
```

Answer Honestly:

How many occurrences of the same “chunk” of code do you **typically** need before you move it to a ”sub-routine”?

Good Code:

- **Reason – able**
- **Readable**
- **Understandable**
- **Changeable**
- Reusable

Good Code:

- **Reason – able**
- **Readable**
- **Understandable**
- **Changeable**
- Reusable
- ***Works correctly***

Why is this hard?
Why do we all have bad code?
500 line constructors?
1000? 10000!? line functions?

Why is this hard?
Why do we all have bad code?
500 line constructors?
1000? 10000!? line functions?

Laziness?

Why is this hard?
Why do we all have bad code?
500 line constructors?
1000? 10000!? line functions?

Laziness?
Focus?

Why is this hard?
Why do we all have bad code?
500 line constructors?
1000? 10000!? line functions?

Laziness?
Too much Focus.

Tony Van Eerd
@tvaneerd

Deconstructivism can be taken too far. There are no bits, only bytes. Only numbers and powers of 2. Don't deconstruct values. #PostModernCpp

RETWEET	LIKES
1	12

8:45 AM - 3 Apr 2017

2 1 12

Tweet your reply

JF Bastien @jfbastien · Apr 3

Replying to @tvaneerd

In C++, there is only destruction†

—

† unless the type is trivially destructible.

1 3

Thomas Rodgers @rodgertq · Apr 3

and ruin

1.7 The C++ memory model

The fundamental storage unit in the C++ memory model is the *byte*…

…is composed of a contiguous sequence of bits

…The least significant bit is called the *low-order bit*; the most significant bit is called the *high-order bit*.

1.7 The C++ memory model

The fundamental storage unit in the C++ memory model is the *byte*…

…is composed of a contiguous sequence of bits

…The least significant bit is called the *low-order bit*; the most significant bit is called the *high-order bit*.

5.11 Bitwise AND operator

…the result is the bitwise AND function of the operands.

1.7 The C++ memory model

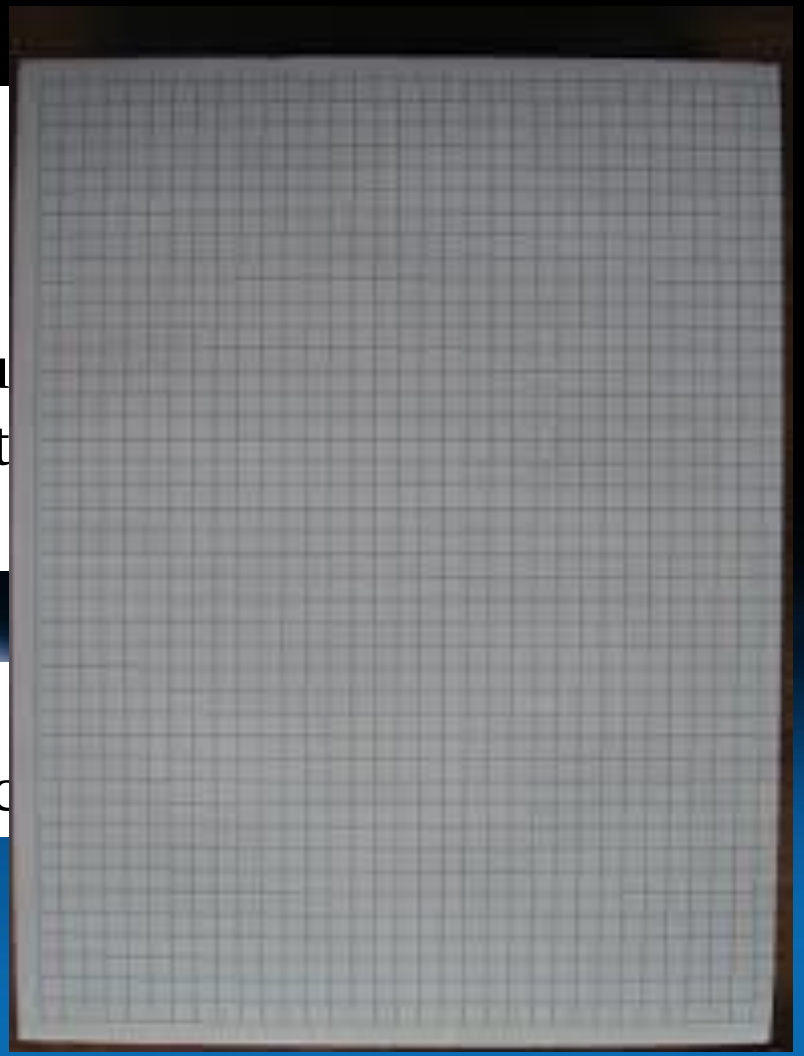
The fundamental storage unit in the
is the *byte*…

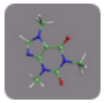
…is composed of a contiguous sequ

…The least significant bit is called t
the most significant bit is called the

5.11 Bitwise AND operator

…the result is the bitwise AND func





Patrice Roy @PatriceRoy1 · Feb 25

Inspiration for #postmoderncpp mabye, what do you think @tvaneerd ?

> **Python Hub** @PythonHub
> Postmodern Error Handling in Python 3.6 journalpanic.com/post/postmoder…

2

Tony Van Eerd @tvaneerd · Feb 25

hmmm, someone who (admits) doesn't understand the power of 'postmodern', sadly. Meditate on 'postmodern error handling' I will.

1

Andrew Pardoe @apardoe · Feb 25

fr.wikipedia.org/wiki/Jean-Fran… Knowledge makes no claims to being original or true, predictions have no relation to reality.

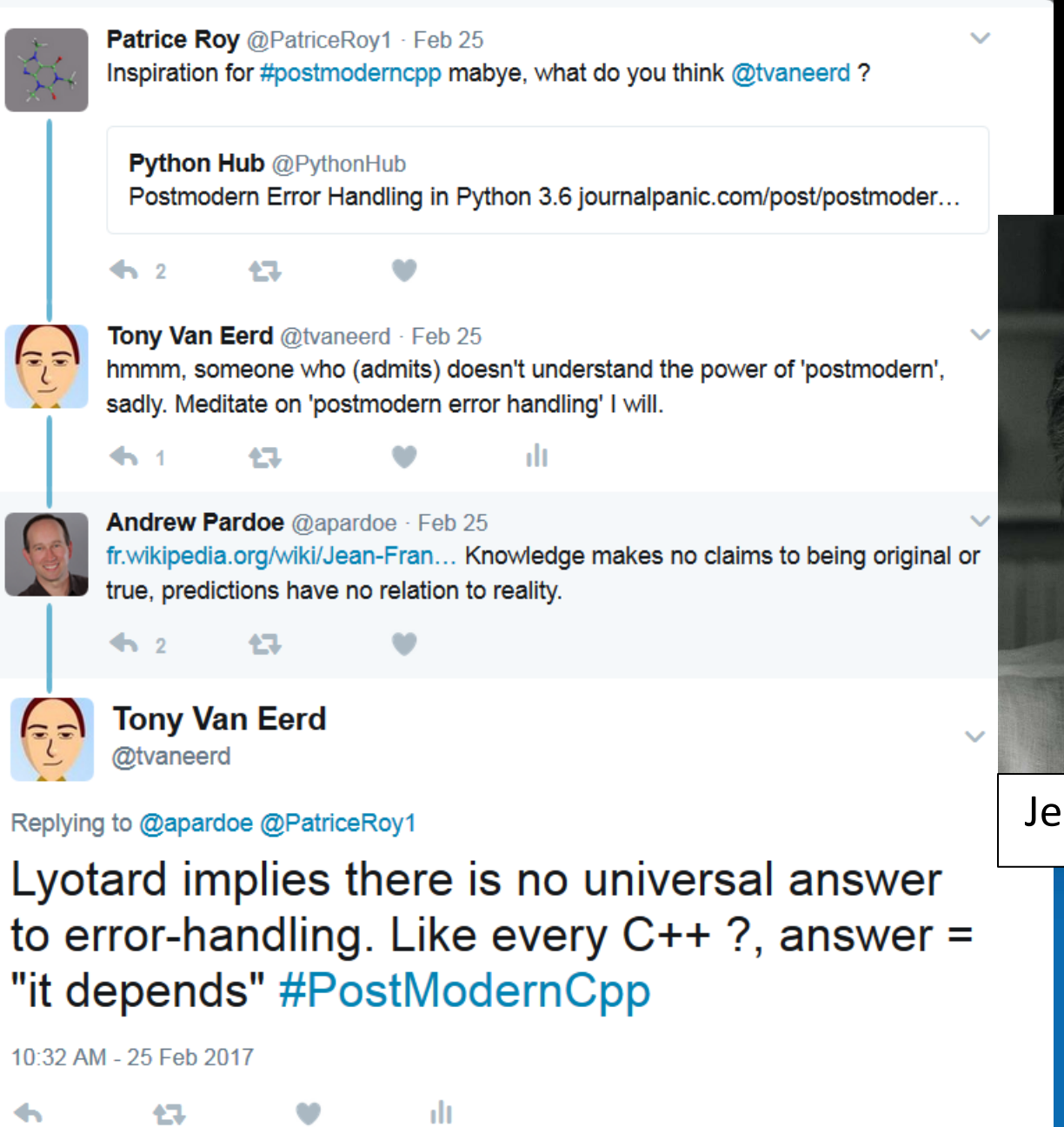
2

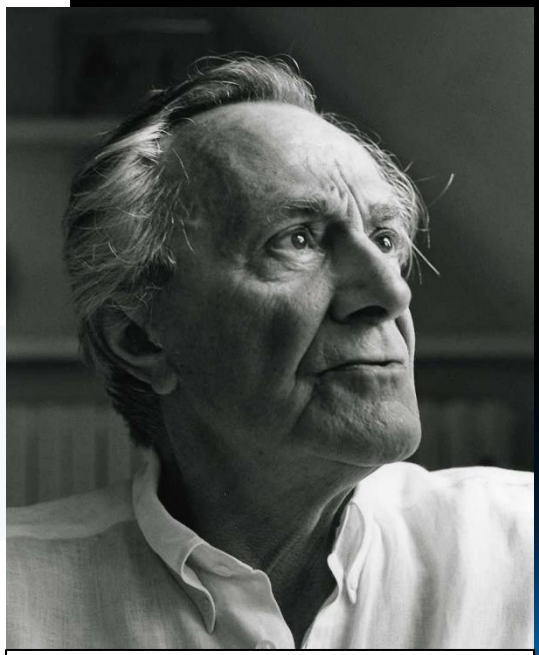
Tony Van Eerd
@tvaneerd

Replying to @apardoe @PatriceRoy1

Lyotard implies there is no universal answer to error-handling. Like every C++ ?, answer = "it depends" #PostModernCpp

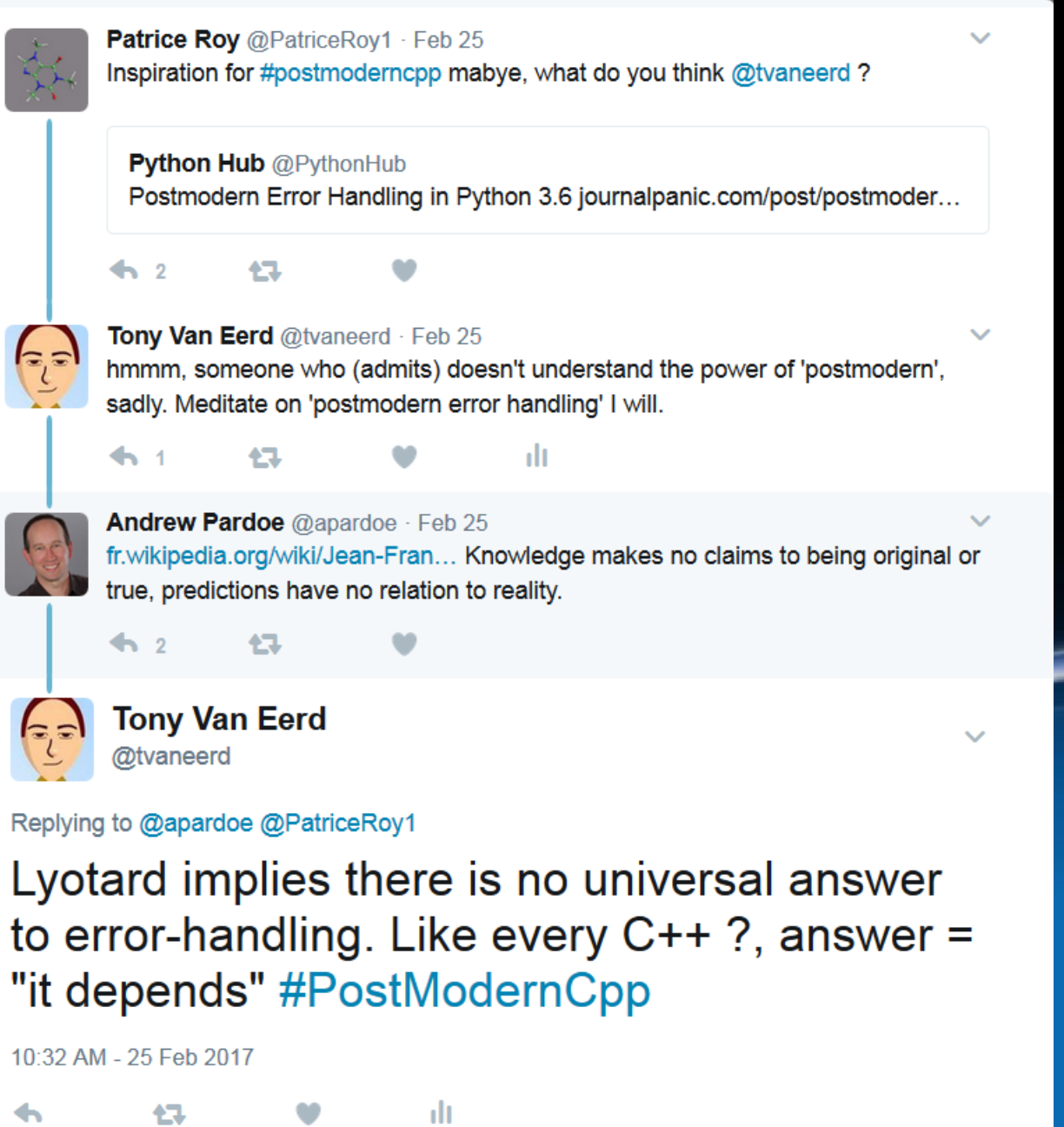
10:32 AM - 25 Feb 2017





Jean-François Lyotard
1924-1998

“It depends”
The curse of being
A multi-paradigm
language



Patrice Roy @PatriceRoy1 · Feb 25
Inspiration for #postmoderncpp mabye, what do you think @tvaneerd ?

Python Hub @PythonHub
Postmodern Error Handling in Python 3.6 journalpanic.com/post/postmoder…

2

Tony Van Eerd @tvaneerd · Feb 25
hmmm, someone who (admits) doesn't understand the power of 'postmodern', sadly. Meditate on 'postmodern error handling' I will.

1

Andrew Pardoe @apardoe · Feb 25
fr.wikipedia.org/wiki/Jean-Fran… Knowledge makes no claims to being original or true, predictions have no relation to reality.

2

Tony Van Eerd
@tvaneerd

Replying to @apardoe @PatriceRoy1

Lyotard implies there is no universal answer to error-handling. Like every C++ ?, answer = "it depends" #PostModernCpp

10:32 AM - 25 Feb 2017

Tony Van Eerd
@tvaneerd

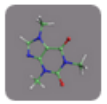
A postmodern audience is aware of conventions used by authors, so error handling code needs to understand to whom it speaks. #PostModernCpp

LIKE
1

7:59 PM - 27 Feb 2017

1 1

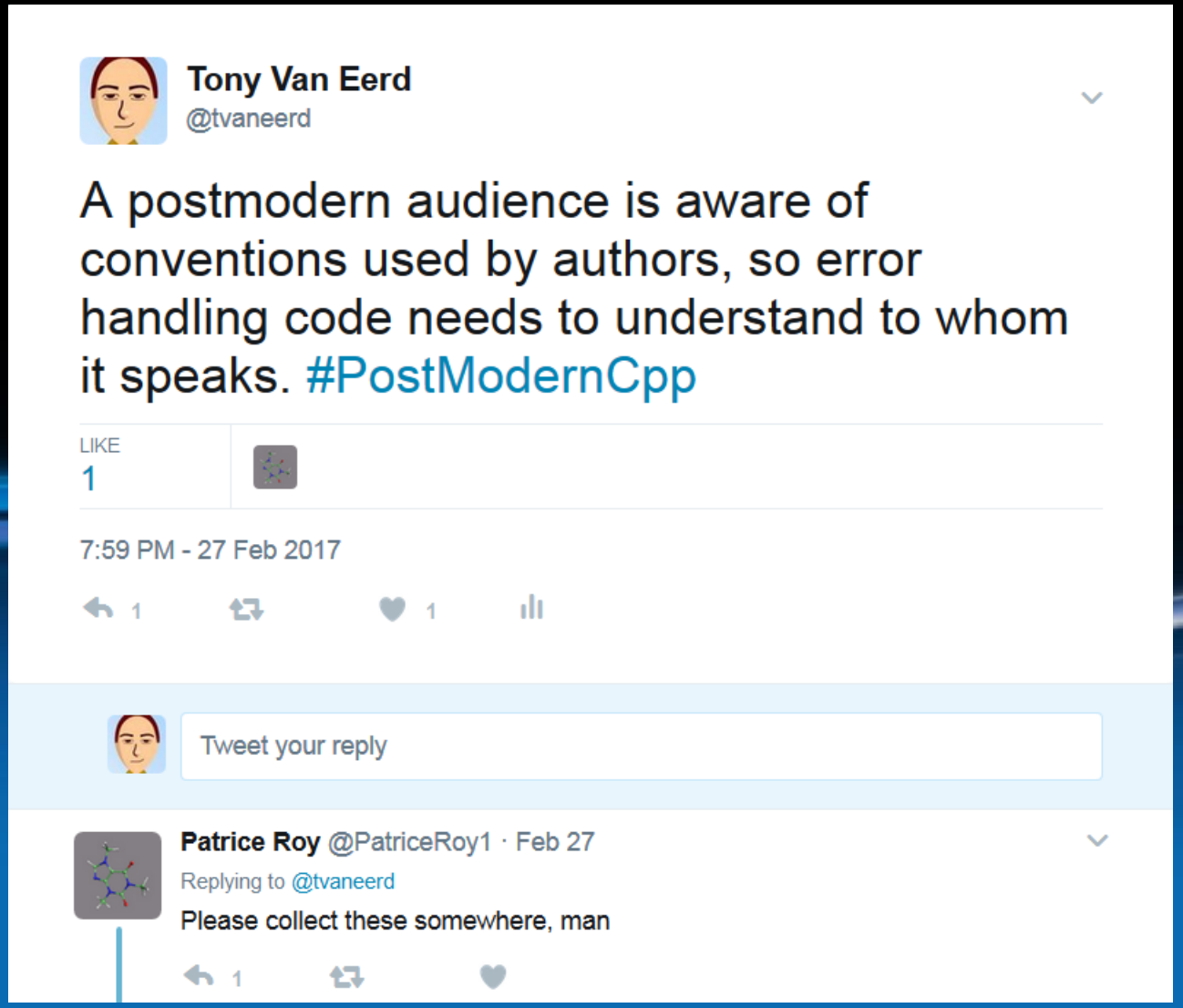
Tweet your reply



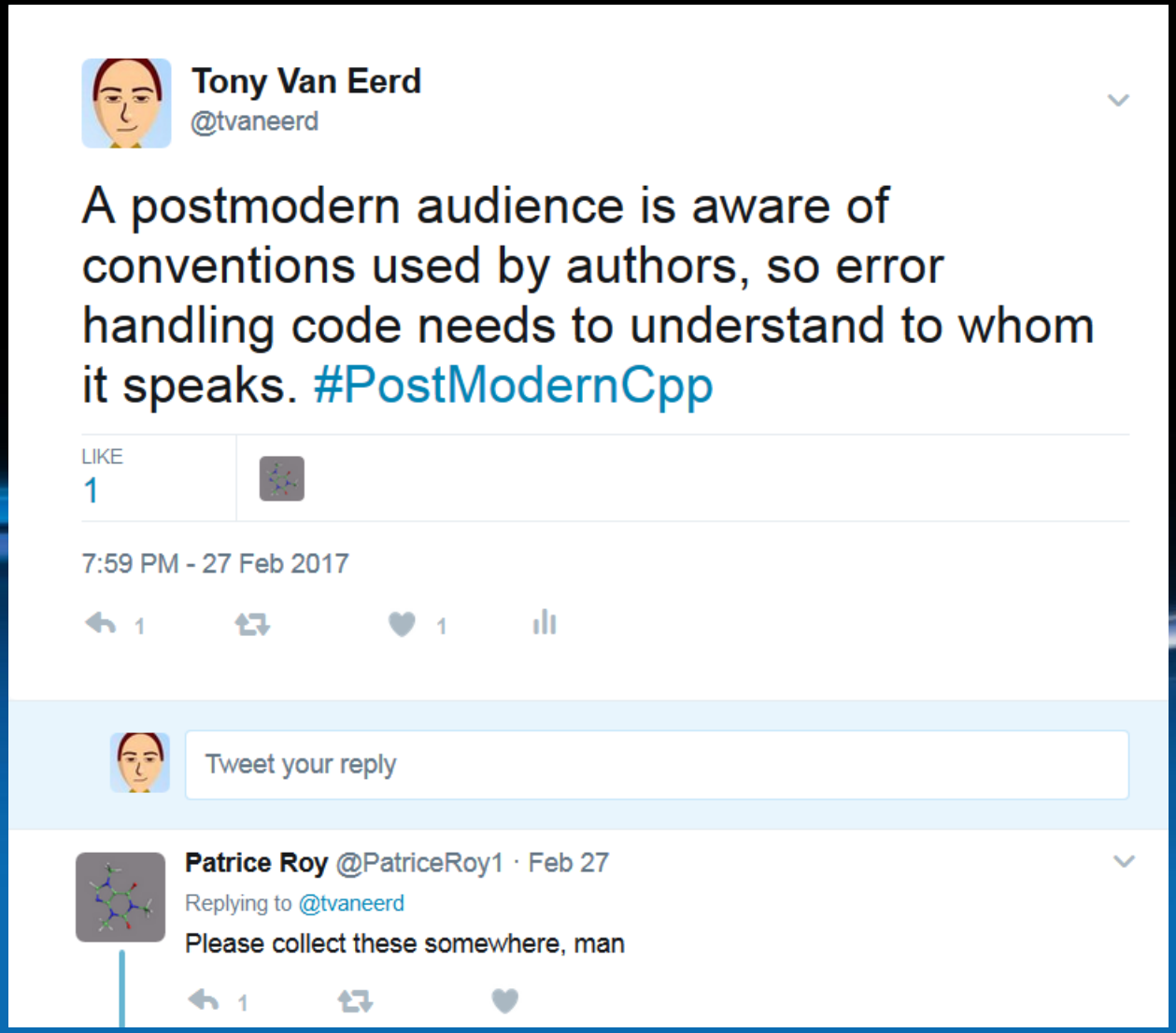
Patrice Roy @PatriceRoy1 · Feb 27
Replying to @tvaneerd
Please collect these somewhere, man

1

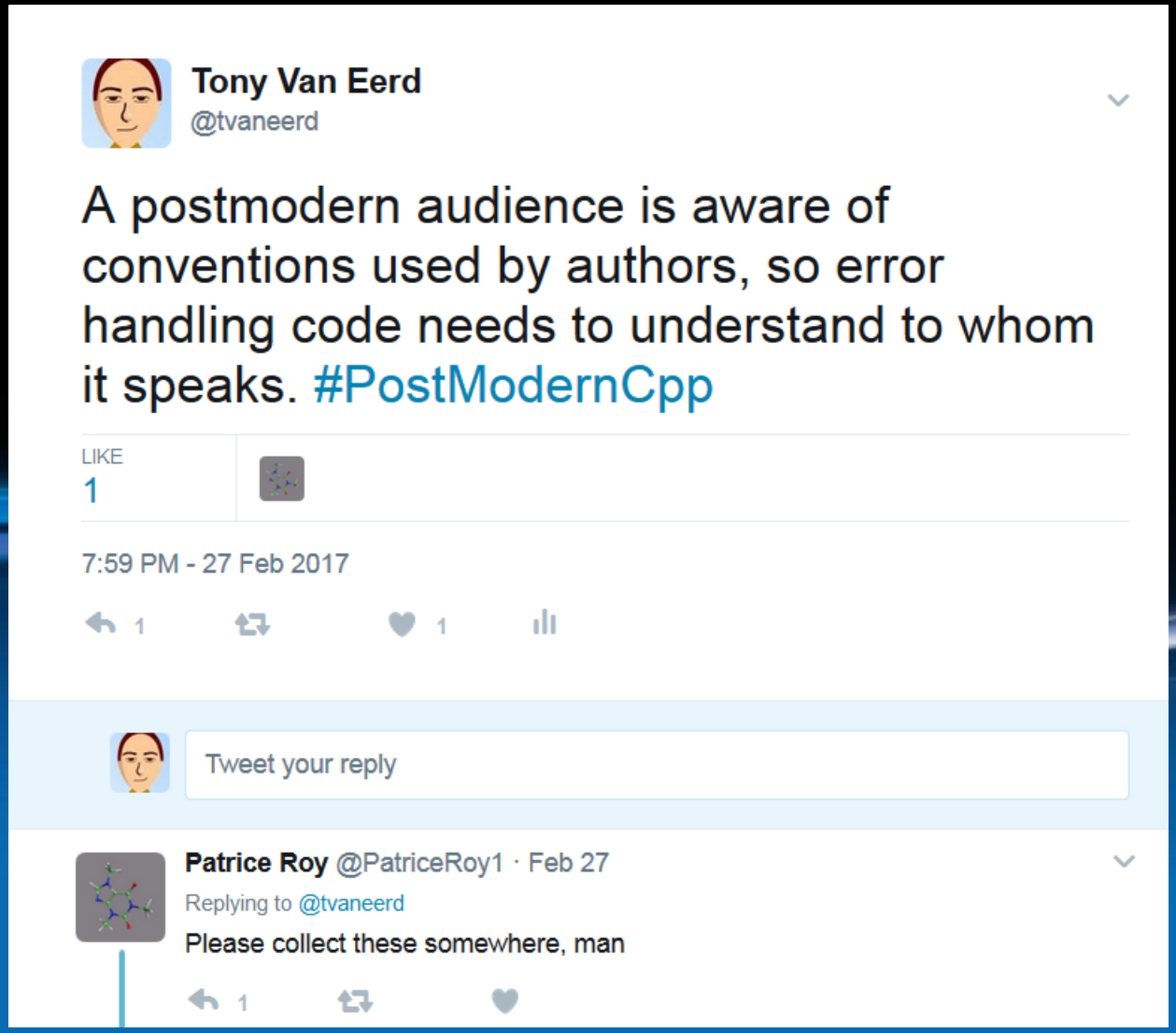
Tropes!


Tony Van Eerd
@tvaneerd
A postmodern audience is aware of conventions used by authors, so error handling code needs to understand to whom it speaks. #PostModernCpp
LIKE
1
7:59 PM - 27 Feb 2017
1
1
Tweet your reply
Patrice Roy @PatriceRoy1 · Feb 27
Replying to @tvaneerd
Please collect these somewhere, man
1

Tropes!
- Exceptions
- Error codes
- Logging
- Termination
- Messages

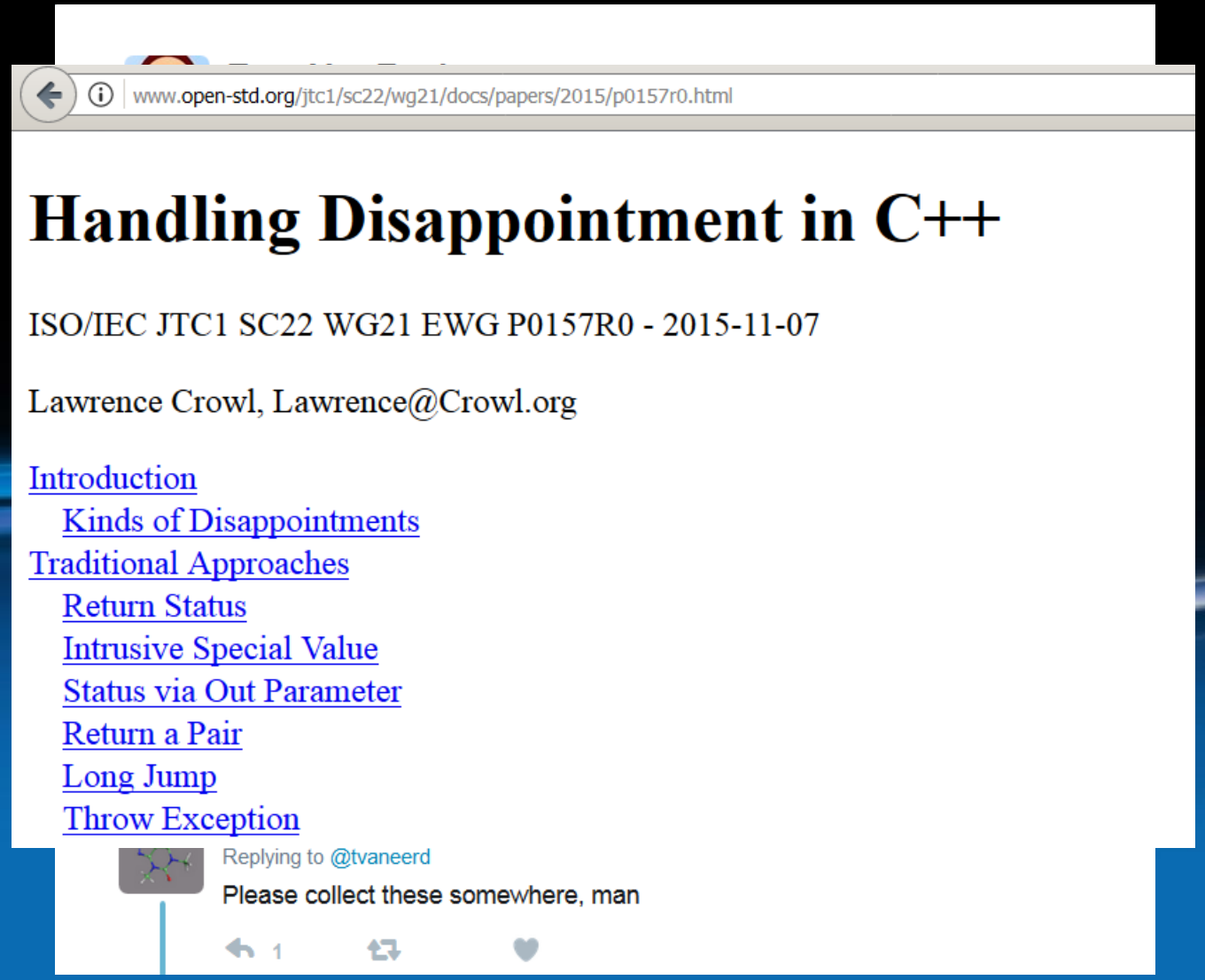


Tropes!
- Exceptions
- Error codes
- Logging
- Termination
- Messages



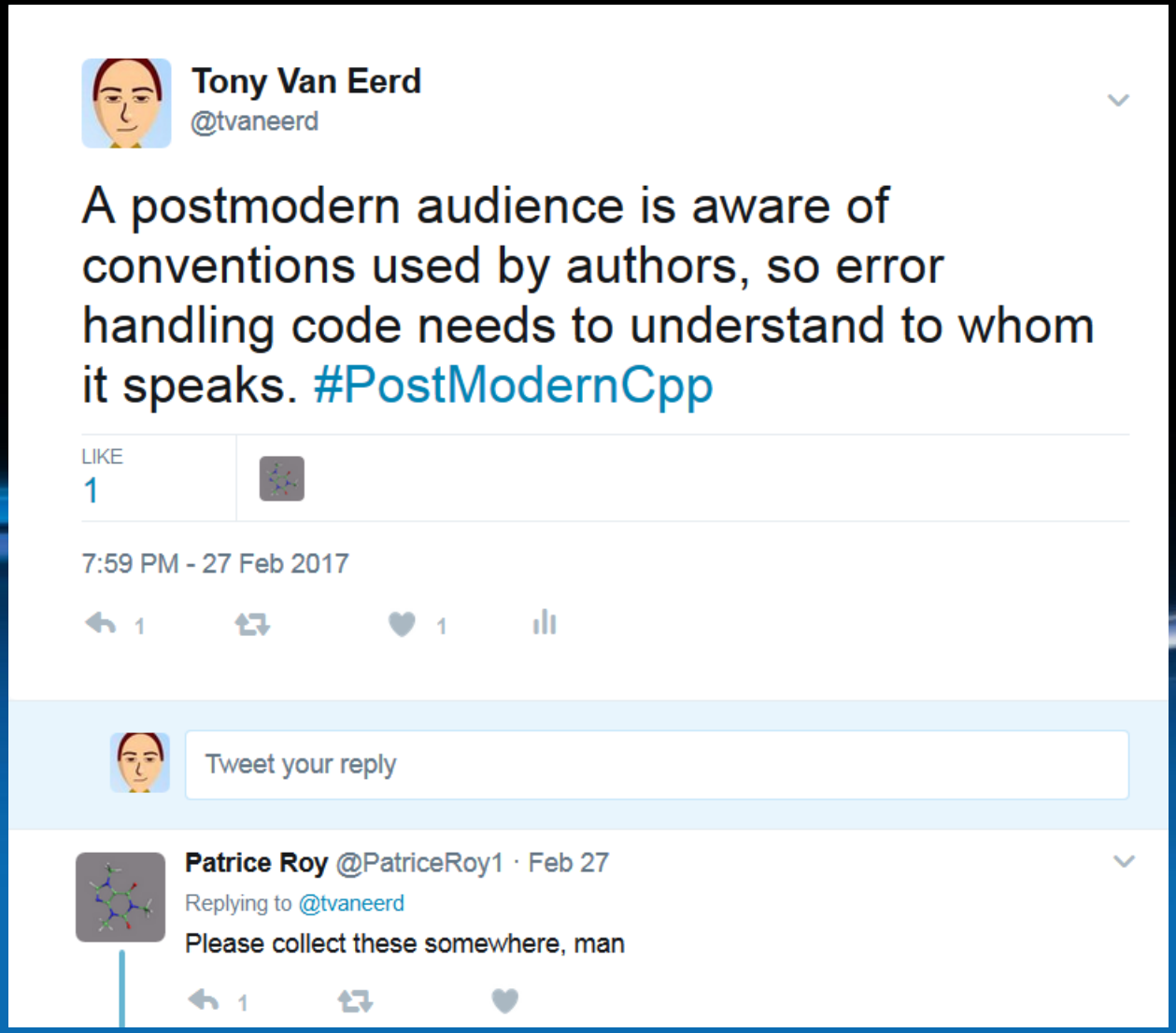
Error Audience
- Library Author
- Calling code
- Calling developer
- End user

Tropes!
- Exceptions
- Error codes
- Logging
- Termination
- Messages



Error Audience
- Library Author
- Calling code
- Calling developer
- End user

Tropes!
- Exceptions
- Error codes
- Logging
- Termination
- Messages



Error Audience
- Library Author
- Calling code
- Calling developer
- End user

Tony Van Eerd
@tvaneerd

If this was the official committee response to an NB comment, would it be rude?: "std::launder is a perfectly cromulent name" #PostModernCpp

LIKES
2

12:53 PM - 30 Jan 2017

3 2

Tweet your reply

Gor Nishanov @GorNishanov · Jan 30
Replying to @tvaneerd
cromulent. adjective. Appearing legitimate but actually being spurious. Was this the designed intent :)

2

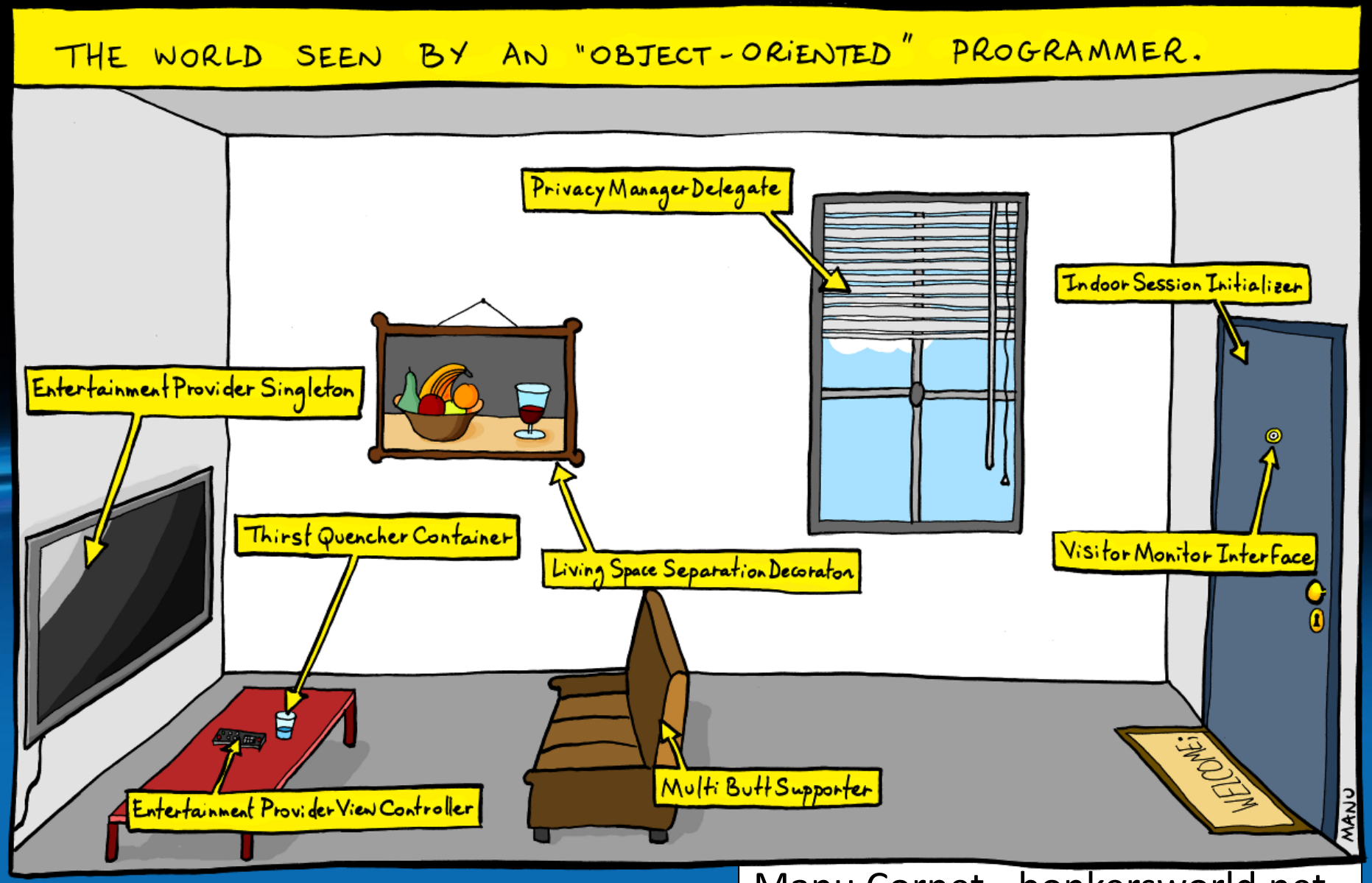
Jason Turner @lefticus · Jan 30
I'm so thankful I can use twitter to embiggen my vocabulary.

On Naming

- Describe the thing in detail – what words did you use?
- Be Consistent
- NOT understanding is better than MISunderstanding
- Co-opt a term?
- Avoid negatives – thus avoiding double negatives
- Avoid spoken ambiguity (or learn to pronounce _, Capitals, etc)
- Avoid verb/noun ambiguity
- Be Concise – conceptually. Avoid sub-concepts.
- By use or by functionality
- Be *Glaringly* Inconsistent
- Determine Essence (and does your API agree)

On Naming

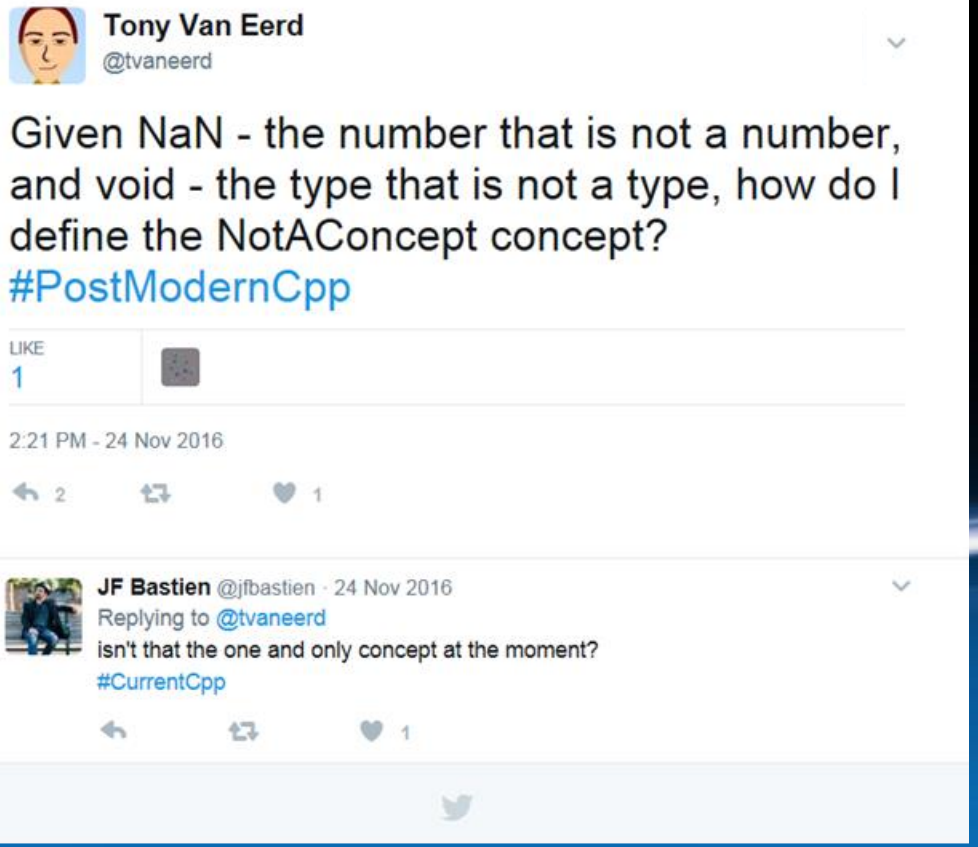
- Describe the thing in detail – what words did you use?
- Be Consistent
- NOT understanding is better than MISunderstanding
- Co-opt a term?
- Avoid negatives – thus avoiding double negatives
- Avoid spoken ambiguity (or learn to pronounce _, Capitals, etc)
- Avoid verb/noun ambiguity
- Be Concise – conceptually. Avoid sub-concepts.
- By use or by functionality
- Be *Glaringly* Inconsistent
- Determine Essence (and does your API agree)

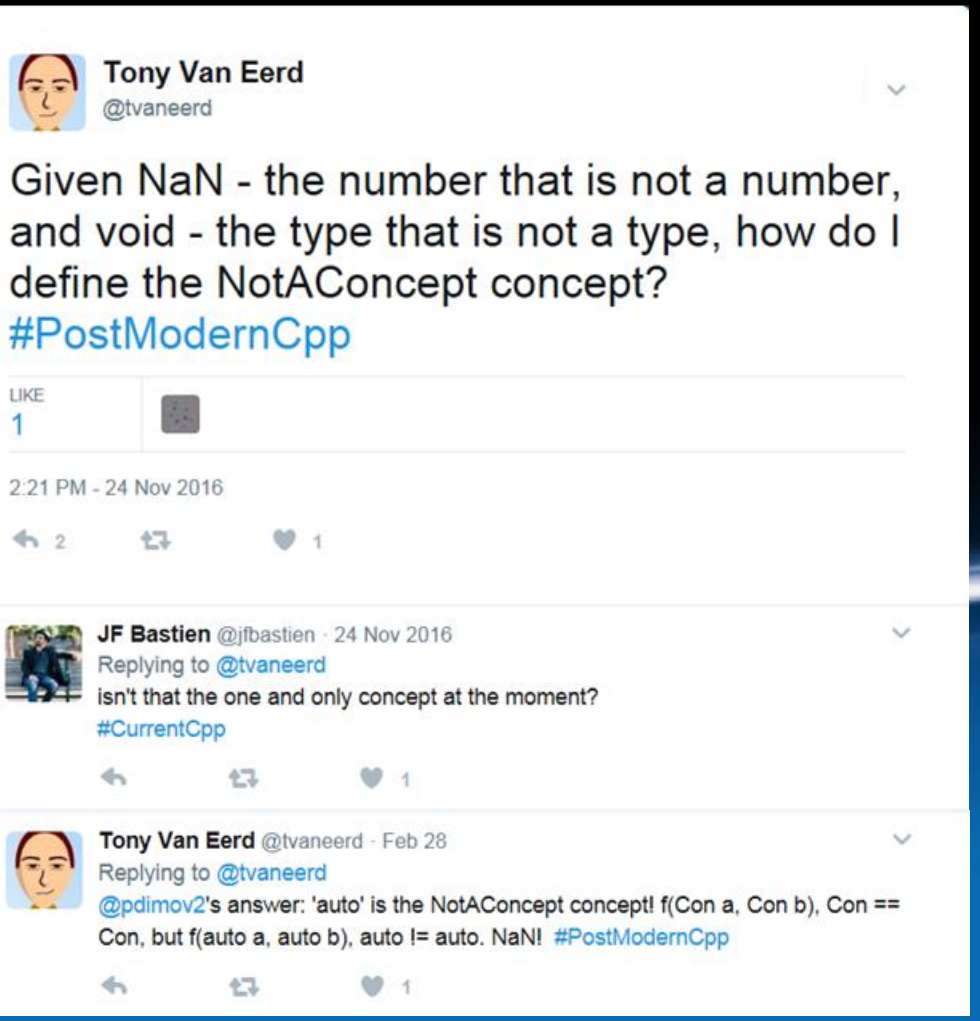


Manu Cornet - bonkersworld.net

On Naming

- Describe the thing in detail – what words did you use?
- Be Consistent
- NOT understanding is better than MISunderstanding
- Co-opt a term?
- Avoid negatives – thus avoiding double negatives
- Avoid spoken ambiguity (or learn to pronounce _, Capitals, etc)
- Avoid verb/noun ambiguity
- Be Concise – conceptually. Avoid sub-concepts.
- By use or by functionality
- Be *Glaringly* Inconsistent
- Determine Essence (and does your API agree)


Tony Van Eerd
@tvaneerd
Given NaN - the number that is not a number, and void - the type that is not a type, how do I define the NotAConcept concept? #PostModernCpp
LIKE
1
2:21 PM - 24 Nov 2016
2
1
JF Bastien @jfbastien · 24 Nov 2016
Replying to @tvaneerd
isn't that the one and only concept at the moment?
#CurrentCpp
1


Tony Van Eerd
@tvaneerd
Given NaN - the number that is not a number, and void - the type that is not a type, how do I define the NotAConcept concept? #PostModernCpp
LIKE
1
2:21 PM - 24 Nov 2016
2
1
JF Bastien @jfbastien · 24 Nov 2016
Replying to @tvaneerd
isn't that the one and only concept at the moment?
#CurrentCpp
1
Tony Van Eerd @tvaneerd · Feb 28
Replying to @tvaneerd
@pdimov2's answer: 'auto' is the NotAConcept concept! f(Con a, Con b), Con == Con, but f(auto a, auto b), auto != auto. NaN! #PostModernCpp
1

```
auto add = [](auto a, auto b) { return a + b; }
Number add(Number a, Number b) { return a + b; }
```

```
auto add = [](auto a, auto b) { return a + b; }
Number add(Number a, Number b) { return a + b; }
```

`Number == Number`, but, `auto != auto` (like `NaN != NaN`)

```
auto add = [](auto a, auto b) { return a + b; }
Number add(Number a, Number b) { return a + b; }

Number add(Number a, Number b) {
    Number c = …;
    …
}
Number add(Number... args) { … }
Number add(Number a, Number b, Number… rest){ … }

class Writeable {…};
void foo(Writeable & a, Writable & b);

void foo(int x, int y);
```

Kathryn Schulz:

On being wrong

TED2011 · 17:51 · **Filmed** Mar 2011

42 subtitle languages

View interactive transcript

“How does it feel to be wrong?”

“How does it feel to be wrong?”

“…it feels like being right.”

Tony Van Eerd
@tvaneerd

What _is_ #PostModernCpp? Find out at C++Now cppnow2017.sched.com/event/A8Iq/pos… CLICKBAIT: You'll never believe what a postmodern smart ptr looks like!

C++Now 2017: Postmodern C++
View more about this event at C++Now 2017
cppnow2017.sched.com

LIKES
2

6:37 PM - 30 Apr 2017

A Proposal for the World's Dumbest Smart Pointer, v4

Document #: WG21 N4282
Date: 2014-11-07
Revises: N3840, N3740, N3514
Project: JTC1.22.32 Programming Language C++

Reply to: Walter E. Brown <webrown.cpp@gmail.com>

Contents

Abstract

This paper proposes **`observer_ptr`**, a (not very) smart pointer type that takes no ownership responsibility for its pointees, i.e., for the objects it observes. As such, it is intended as a near drop-in replacement for raw pointer types, with the advantage that, as a vocabulary type, it indicates its intended use without need for detailed analysis by code readers.

I'm an observer in these matters, not a participant.
— MARSHALL MCLUHAN

Sometimes it is the quiet observer who sees the most.
— KATHRYN L. NELSON

Tony Van Eerd
@tvaneerd

Two readers reading the same source may see different values due to what each read and wrote previously. #PostModernCpp #MemoryModel #lockfr

RETWEET 1 LIKES 2

8:28 AM - 19 Oct 2016

1 2

Conclusions

Conclusions

“I'm seeing lots of random thoughts and mini-topics, but not a coherent narrative”

“The scope of this talk looks quite vague”

“This topic has the potential to be either very good or very bad…
…I would feel better if there were a bit more to the abstract ”

Conclusions

“I'm seeing lots of random thoughts and mini-topics, but not a coherent narrative”

“The scope of this talk looks quite vague”

“This topic has the potential to be either very good or very bad…
…I would feel better if there were a bit more to the abstract ”

“The outline appears to have _context_ as the central driving theme”

Conclusions

- Context
- Communication
- Constraints
- Multi-level Thinking
- Assumptions
- Awareness
- Ambiguity
- Patterns/Paradigms

Conclusions

- Context
- Communi
- Constraint
- Multi-leve

s

radigms

Conclusions

- Context
- Communication
  - **Compiler**
  - **Computer**
  - **End User**
  - **Developers**
- Constraints
- Multi-level Thinking
- Assumptions
- Awareness
- Ambiguity
- Patterns/Paradigms

Conclusions

- Context
- Communication
  - **Compiler**
  - **Computer**
  - **End User**
  - **Developers**
- Constraints
- Multi-level Thinking
- Assumptions
- Awareness
- Ambiguity
- Patterns/Paradigms

Conclusions

- Context
- Communication
- Constraints
- Multi-level Thinking
- Assumptions
- Awareness
- Ambiguity
- Patterns/Paradigms

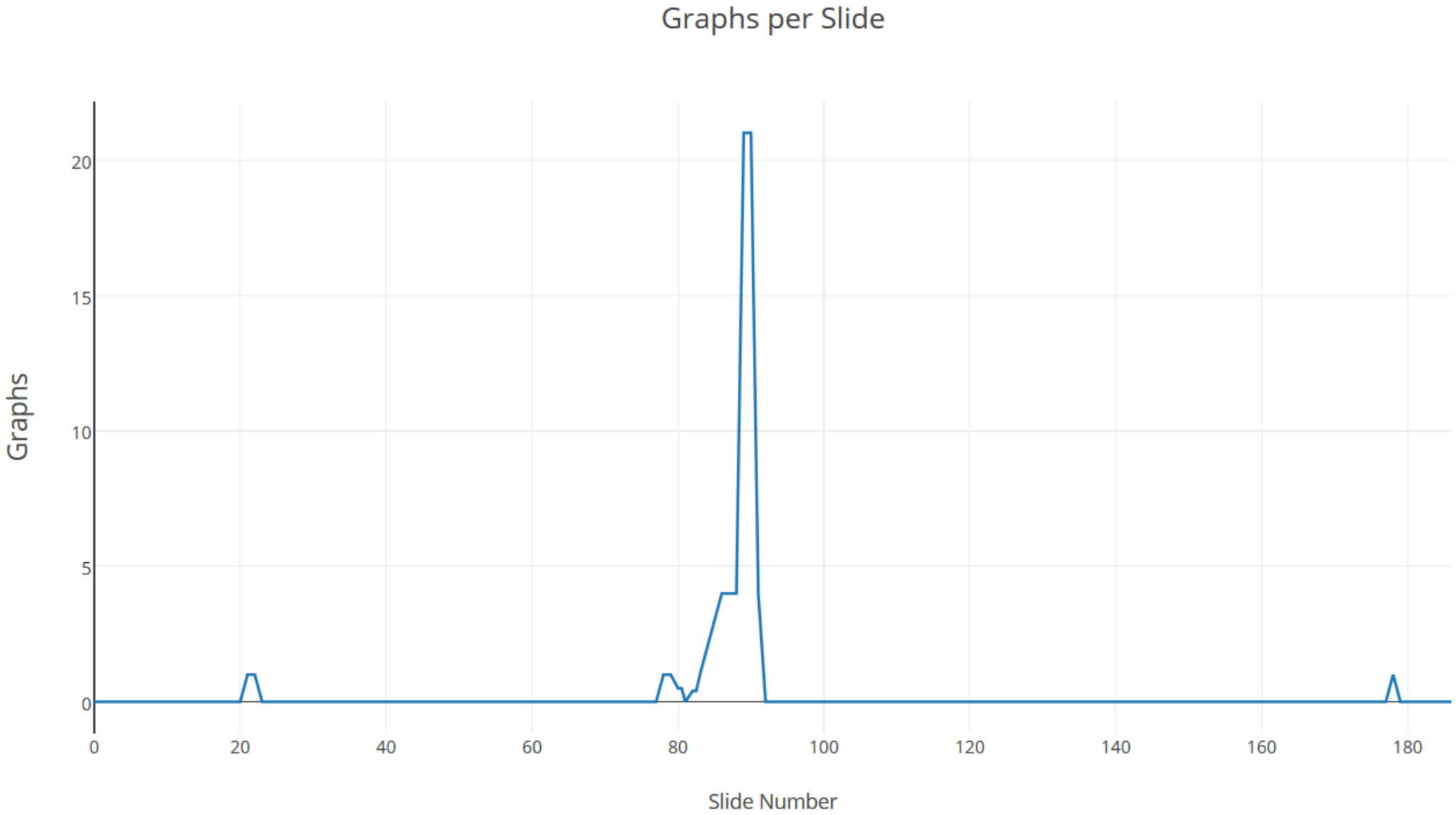
Conclusions

- Context
- Communication
- Constraints
- Multi-level Thinking
- Assumptions
- Awareness
- Ambiguity
- Patterns/Paradigms

Headline:
“Human Brain Mimicking AI Platform Conquers Media Summarization, Looks to Expand”

Conclusions

- Context
- Communication
- Constraints
- Multi-level Thinking
- Assumptions
- Awareness
- Ambiguity
- Patterns/Paradigms


Graphs per Slide
Graphs
0
5
10
15
20
0
20
40
60
80
100
120
140
160
180
Slide Number

Bonus Slides

Tony Van Eerd

C++Now 2017

C++ Koans

Tony Van Eerd

C++Now 2017

Tony Van Eerd
@tvaneerd

"Master it says //must do X first!
but why?"
"Check Git"
After a week finding nothing in the archives,
the student was enlightened
#CppKoan

LIKE
1

12:40 PM - 29 Apr 2017

1 1

Tweet your reply

Jason Rice @JasonRice_ · Apr 29

Replying to @tvaneerd

I might get this in a week. 😝

2

Tony Van Eerd
@tvaneerd

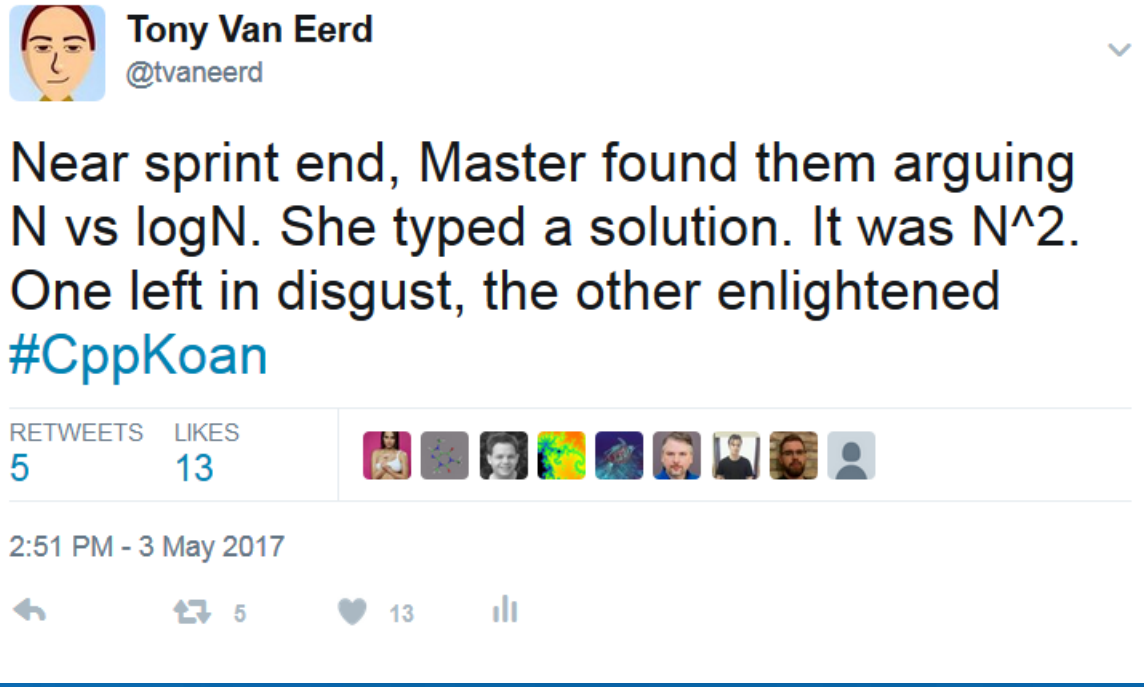
"LockFree Master we need a queue"
He searched the codebase, then typed
template<Regular T>
using ThreadSafeQueue =
MutexedQueue<T>;
#CppKoan

1:04 PM - 16 May 2017

Tony Van Eerd
@tvaneerd

"Master, explain the Rule of Zero."
"Study the ptrs of Masters Hinnant and Dimov."
"But they have destructors..."
"As they should."
#CppKoan

6:35 PM - 26 Apr 2017


Tony Van Eerd
@tvaneerd
Near sprint end, Master found them arguing N vs logN. She typed a solution. It was N^2. One left in disgust, the other enlightened #CppKoan
RETWEETS
5
LIKES
13
2:51 PM - 3 May 2017
5
13

Tony Van Eerd

C++Now 2017

CHRISTIE