

Greetings Fellow D-velopers!

This presentation is available online at

<http://flowvel.la/wonk>



confessions of a C++ wonk

why D is very, very good for me!

chuck allison



Definition (from dictionary.com)

wonk 

[wɒŋk]

Spell

Syllables

[Examples](#)

[Word Origin](#)

noun, *Slang*.

1. a student who spends much time studying and has little or no social life; grind.
2. a stupid, boring, or unattractive person.
3. a person who studies a subject or issue in an excessively assiduous and thorough manner:
a policy wonk.

What's

KNOW

spelled backward?

WONK

The Language Landscape

Let's visit tiobe.com



how did we get
here?

I was there...

chico escuela

“Baseball ... been berry, berry good to me.”



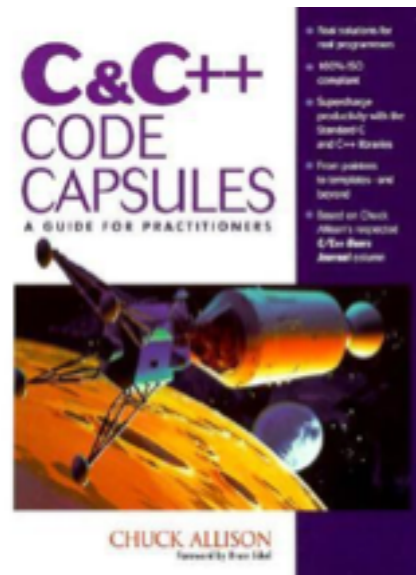
(aka Garrett Morris)

“C++ has been very, very good to me.”

C++

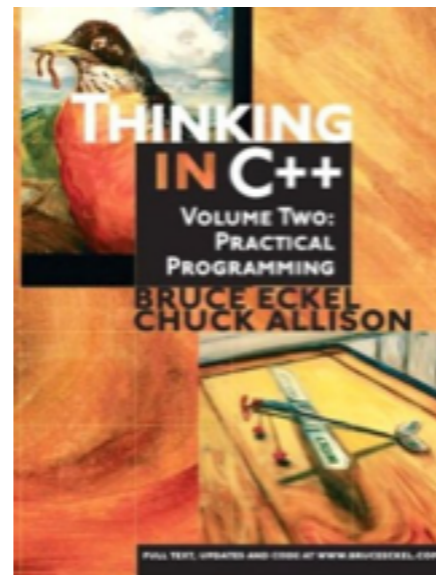


Books



1998

English
Chinese



2004

English
Chinese
Czech
Polish



2010

English
Chinese
Czech
Russian
Japanese
Korean

credit where credit is due



Adopted Organically (Like C)

C++ took OOP to “the masses”

http://padlet.com/chuck_allison/cpp_good

(double-click)



My Two Cents



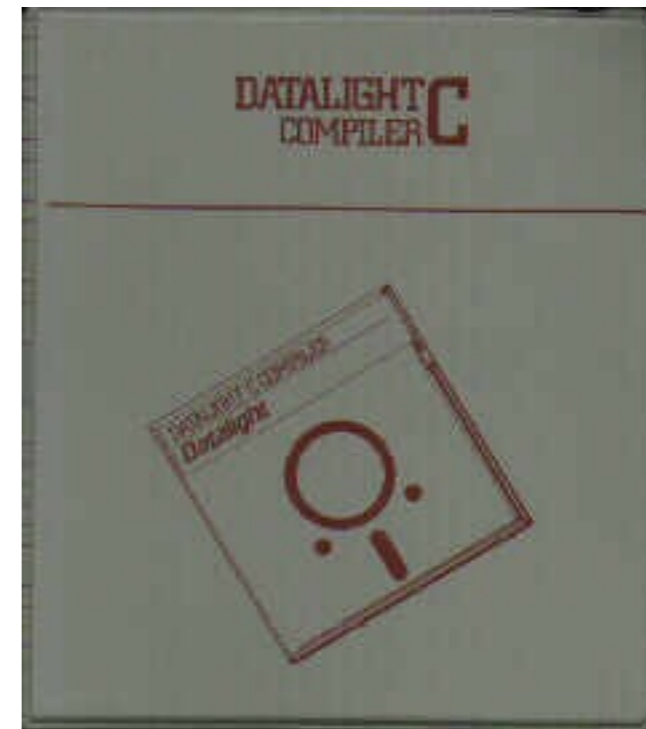
Pre-History



1975 – FORTRAN

1984 – Mark Williams C

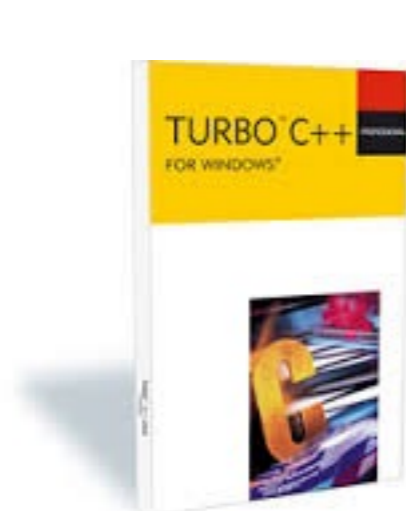
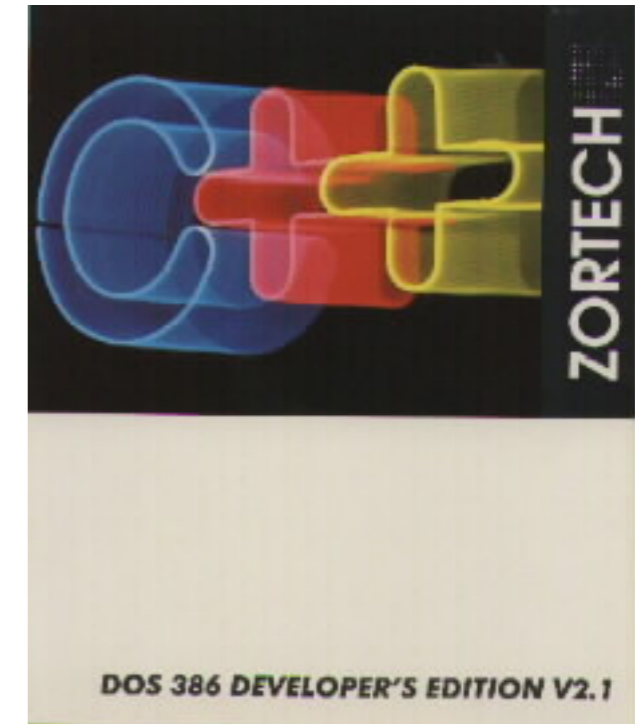
1987 – Datalight C



1989–1990

Early adopter of Zortech C++

Wrote 5 chapters of the Turbo-Borland C++ documentation



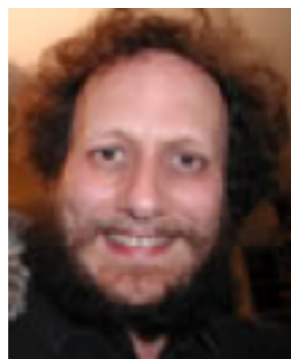
1991

ANSI Committee J16 “Programming Language C++”



Present at first technical meeting
(March 1991, Nashua, NH)

- represented 3 organizations
- served actively 10 years



Designed/implemented **std::bitset**

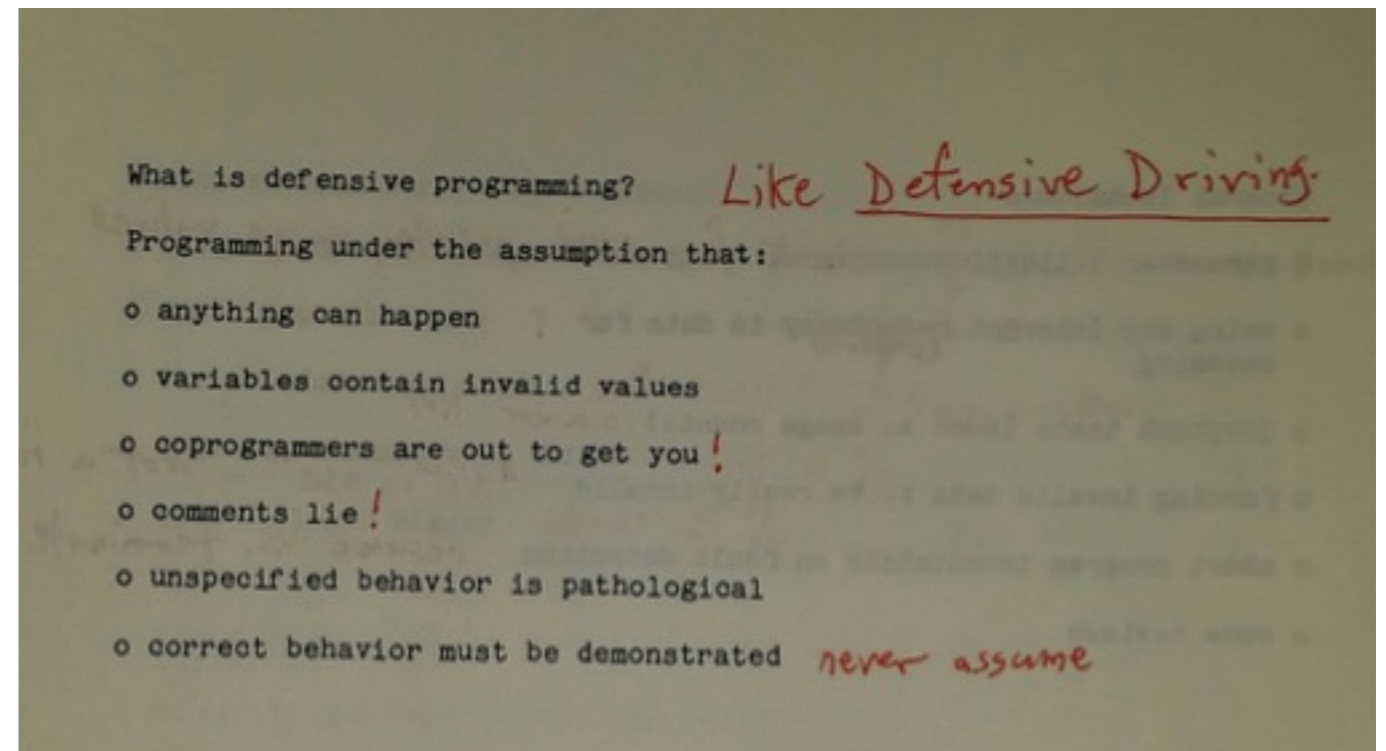
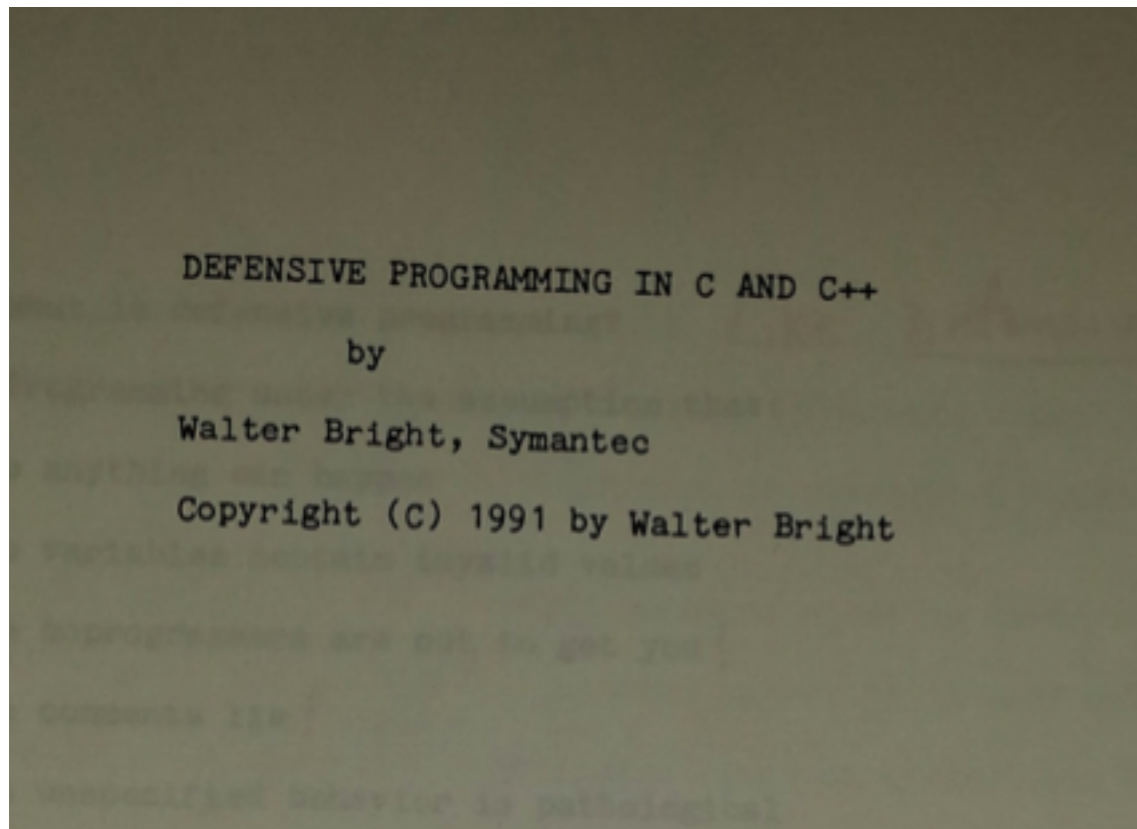
And **boost::dynamic_bitset**

- with Jeremy Siek

1991

C++ World

Met this fellow named
Walter Bright



1992–2003



“Code Capsules” Columnist
1992–1994

Consulting Editor, 1996–2001

Senior Editor, 2001–2003



2003–2008

Founder and Editor



Editor
Chuck Allison

Enforcing Code Feature Requirements in C++
by Scott Meyers, September 23, 2008,  16 comments

<http://www.artima.com/cppsource>

Advisory Board

David Abrahams	Andrew Koenig
Andrei Alexandrescu	Angelika Langer
Matthew Austern	Nathan Myers
Walter Bright	Eric Niebler
Steve Clamage	Thorsten Ottosen
Greg Colvin	Thomas Plum
Jim Coplien	Dan Saks
Stephen Dewhurst	Jeremy Siek
Neil Harrison	Bjarne Stroustrup
Kevlin Henney	Herb Sutter
Howard Hinnant	Matthew Wilson



Why Java Succeeded

Novelty of Internet

Portability of VM

Garbage Collection

No Raw Pointers!

SD WEST 2007 MARCH 19-23, 2007
SANTA CLARA CONVENTION CENTER
SANTA CLARA, CA

20 CELEBRATING YEARS!

OVER 200 SESSIONS IN 14 IN-DEPTH TRACKS

- Business of Software
- C++
- Java
- .NET
- Ruby **NEW**
- Modeling & Design
- People, Projects & Methods
- Requirements & Analysis
- Security
- Testing & Quality
- Web 2.0 **NEW**
- Web Services/SOA
- Windows Vista **NEW**
- XML

BACK BY POPULAR DEMAND:
STROUSTRUP & SUTTER ON C++
Join C++ creator Bjarne Stroustrup and renowned C++ expert Herb Sutter for an in-depth, two-day tutorial on C++.

PLUS: Keynotes, Expo, Birds-of-a-Feathers, Panels, Case Studies, Roundtables, Parties and Special Events

ALL THE KNOWLEDGE YOU NEED

REGISTER TODAY AT WWW.SDEXPO.COM

SUPER EARLY BIRD DISCOUNT
Register by JAN 19
SAVE UP TO \$500

EARLY BIRD DISCOUNT
Register by FEB 23
SAVE UP TO \$300

2004

The D Programming Language

by Walter Bright

D is an advanced systems programming language. It is designed to appeal to C and C++ programmers who need a more powerful language that has a much lower complexity and hence is easier to master. D is multiparadigm, and looks and feels very much like C and C++. It offers opportunities for advanced compilers to generate more efficient code than is possible for C/C++, while supporting facilities that reduce the probability of program bugs.

Why D?

Refactoring

C++ has been around for 20 years now. C++ has largely succeeded in adding enormous capability to C while retaining backwards compatibility with it. But with 20 years experience comes the opportunity to reflect on how one might engineer a language that retains C++'s strengths, add modern features, and remove its weaknesses and more troublesome aspects.

Difficulty in adding modern new features

The longer a language has been evolving, the harder it gets to add new features. Each new feature adds an unanticipated layer on top of old ones, in a way that no legacy code breaks. Eventually, it takes forever to add an insignificant improvement. The C++ 'export' is an extreme example of this effect, taking a reported 3 man years to implement and delivering little apparent benefit. A more mundane indication of this problem is C++ was standardized 5 years ago and just now conformant compilers are emerging.

While C++ is pioneering generic programming practice, it lags behind in other modern techniques such as design by contract, modules, automated testing, and automatic memory management. It's very difficult to add these while still supporting legacy code.

Brief Tour

D looks a great deal like C and C++, so much so that the canonical hello world program is nearly identical:

```
import std.c.stdio;

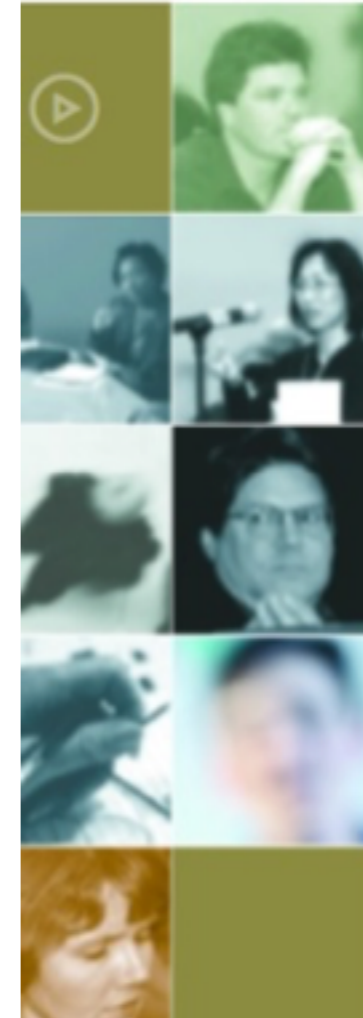
int main()
{
    printf("Hello world!\n");
    return 0;
}
```

Look and feel is very much like C and C++

Many years ago in grammar school, we were shown a film about a researcher who wore special goggles that turned the world upside down. He wore them continuously such that his brain never saw the world right side up. After 2 weeks, his brain suddenly righted that upside down view. Then, the researcher took the goggles off. The film darkly warned the viewer to not try that ourselves!

I am so, so used to C/C++ syntax that I feel like that poor guy when faced with a new and improved language that also turns the syntax inside out (or so it looks to me). Frankly, I rarely give such languages a chance even when the feature set looks intriguing. D doesn't take that route. Its syntax is as comfortable to C/C++ programmers as an old shoe. Functions, statements, expressions, operator precedence, integral promotions, it's all there pretty much unchanged. The world is right side up, it's just got brighter colors and sharper focus!

SPEAKERS



[A]

Steve Adolph
Andrei Alexandrescu
Chuck Allison
Scott Ambler
Jennitta Andrea
David Astels

[B]

Scott Bain
Jonathan Baker
Tracy Bialik
Ron Bodkin
Grady Booch
Toufic Boubez
Walter Bright
Kevin Bryant



why is D Very Good for you?

http://padlet.com/chuck_allison/d_good

why D is very, very

good for me!

All D-developers Love...

D compiles to *native code*

(Optional) **G**arbage **C**ollection

Module System

Slices

 Associative Arrays

static if

Universal **F**unction **C**all **S**yntax 

Compile-**T**ime **F**unction **E**valuation

Mixins (string and template)

unittest

Contract Programming

debug

Other Cool Language Features

 Slices

Delegates & Decorators 

 Array-wise operations

Delegates & Bound Methods 

 **lazy** evaluation

shared 

The Python Library



Batteries Included

- The Python standard library is very extensive
 - regular expressions, codecs
 - date and time, collections, threads and mutexes
 - OS and shell level functions (mv, rm, ls)
 - Support for SQLite and Berkley databases
 - zlib, gzip, bz2, tarfile, csv, xml, md5, sha
 - logging, subprocess, email, json
 - httpplib, imaplib, nntplib, smtplib
 - and much, much more

“Batteries Included”

The D Library



“Nuclear Reactor Included”

Cool Library Features

 **scoped**

memoize 

 **partial** fun evaluation

Fibers 

 Discriminated unions

Ranges 

D-veloper Tools & Resources



rdmd

dlang.org

dub

vibe.d

Xamarin Studio (Mono-D, Mac/Linux)

Books: Alexandrescu, Çehreli, Ruppe

http://padlet.com/chuck_allison/d_tools

more to
come!

Erich Gubler

Friday, 3:30pm