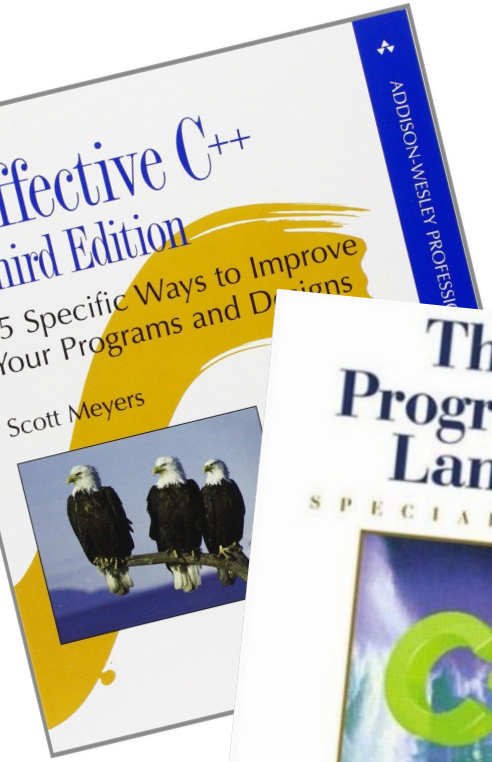
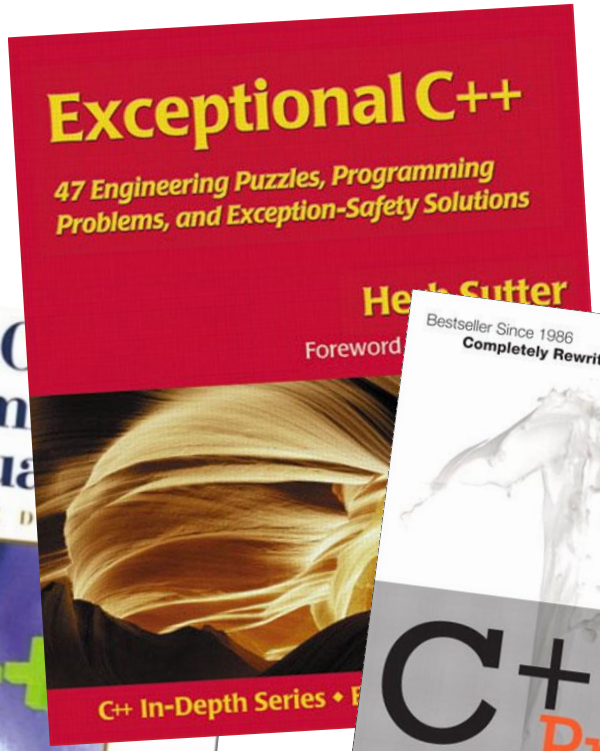
cppreference.com

a brief history of things

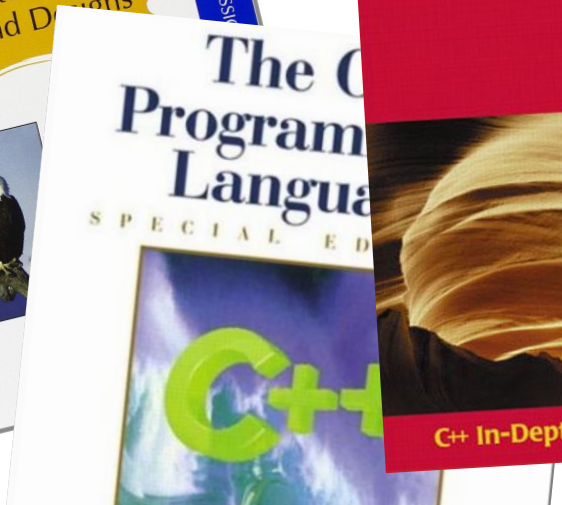
C++ Documentation Today

accu
professionalism in programming





C++ now



C++
cppreference.com

Meeting C++

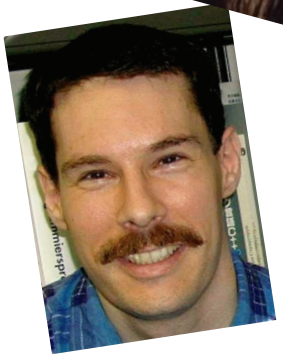
C++ Documentation Today

2000

C++ Documentation Today

C++ Documentation Today

C++ Documentation Today



C++ Documentation Today

C++ Documentation Today

C with Classes

“There was never a C++ paper design: design, documentation, and implementation went on simultaneously...**C++ evolved...through discussions between the author and his friends and colleagues**.”

Early Documentation

1980 Bell Labs internal tech report

1982 “Classes: An Abstract Data Type Facility for the C Language”

1984 “The C++ Reference Manual”

1986 “The C++ Programming Language”

Early Documentation

“Usually, a tutorial was written somewhere along the way. **Writing a tutorial was considered an essential design tool**, because if a feature cannot be explained simply, the burden of supporting it will be too great. This point was never far from my mind because during the early years I *was* the support organization.”

Beginning Standardization

1989 ANSI X3J16 meeting

1990 “The Annotated C++ Reference Manual”

1991 ISO WG21 meeting

1994 ANSI/ISO Committee Draft registered

1998 C++98 ISO standardization

Beginning Standardization

“Only after C++ was an established language did more conventional organizational structures emerge and even then I was officially in change of the reference manual and had the final say over what went into it until that task was handed over to the ANSI C++ committee in early 1990.”

Beginning Standardization

“Sometime in 1988 it became clear that C++ would eventually have to be standardized… Clearly, an effort had to be made to write a more precise and comprehensive definition of the language.”

Beginning Standardization

About the 1990 ARM: “...making an improved reference manual wasn’t something that could be done by one person (me) in private. **Input and feedback from the C++ community was needed.**”

Beginning Standardization

“Newsgroups and bulletin boards such as comp.lang.c++...produced tens of thousands of messages over the years to the delight and despair of their readers.”

Standardization is Great!

1998: good number of books, journals, groups

...but not necessarily that accessible

outsiders? newbies? poor people?

dot-com boom


LET ME INTRODUCE YOU
TO THE INTERNET

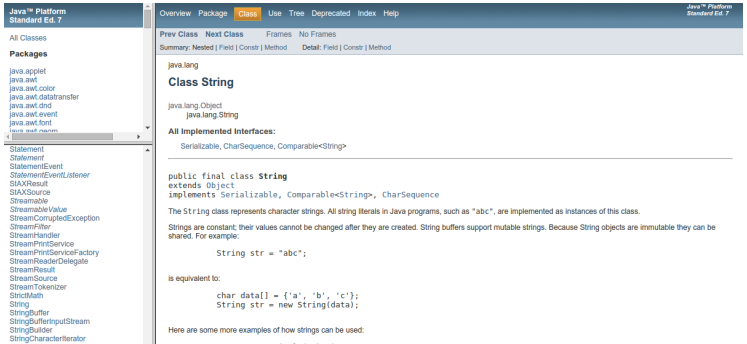
Motivation

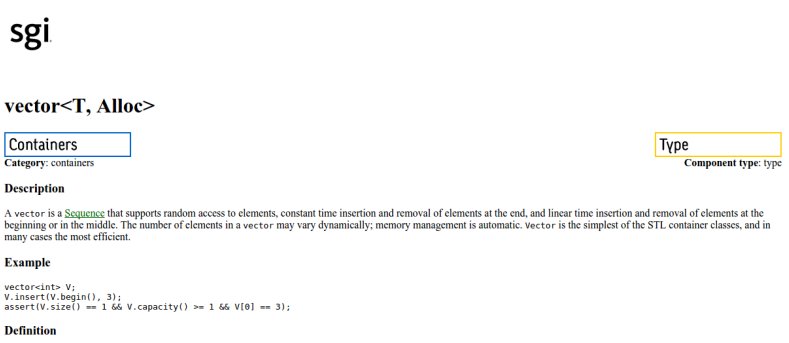
sun’s java docs were useful

SGI’s STL ref was a little intimidating

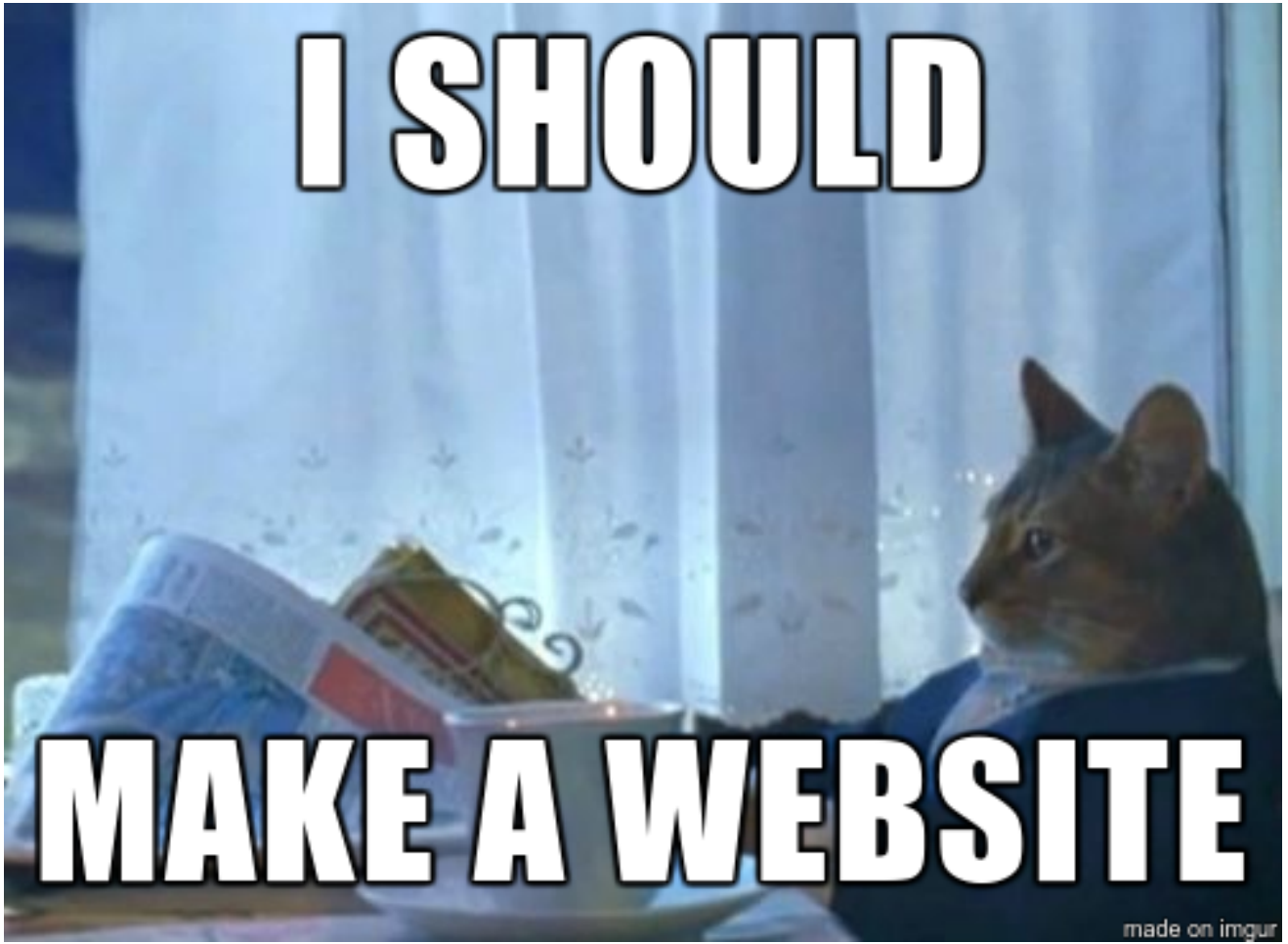
an underserved niche?

too much time on my hands?





2000



Target User

make it useful to the average programmer

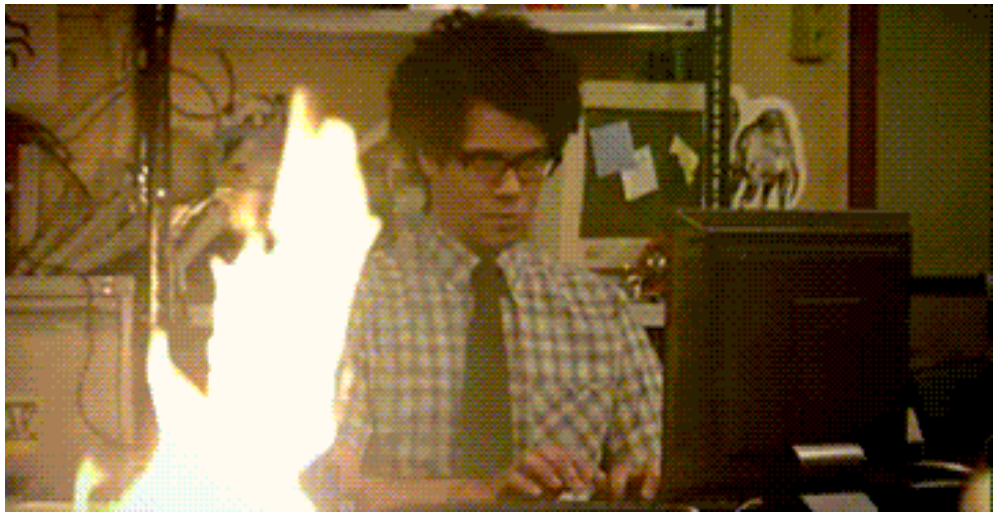
time-constrained

not necessarily a language lawyer

might have other things on their mind

Target User

make it useful to the average programmer



Goals

quick, simple, minimalist

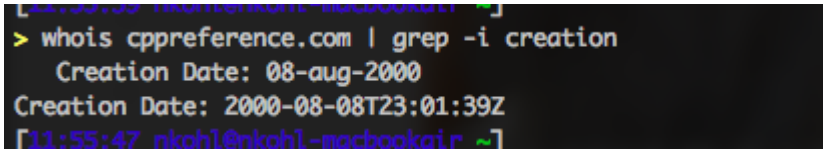
easy-to-understand summaries

examples

rich content when needed, crosslinking

First Steps

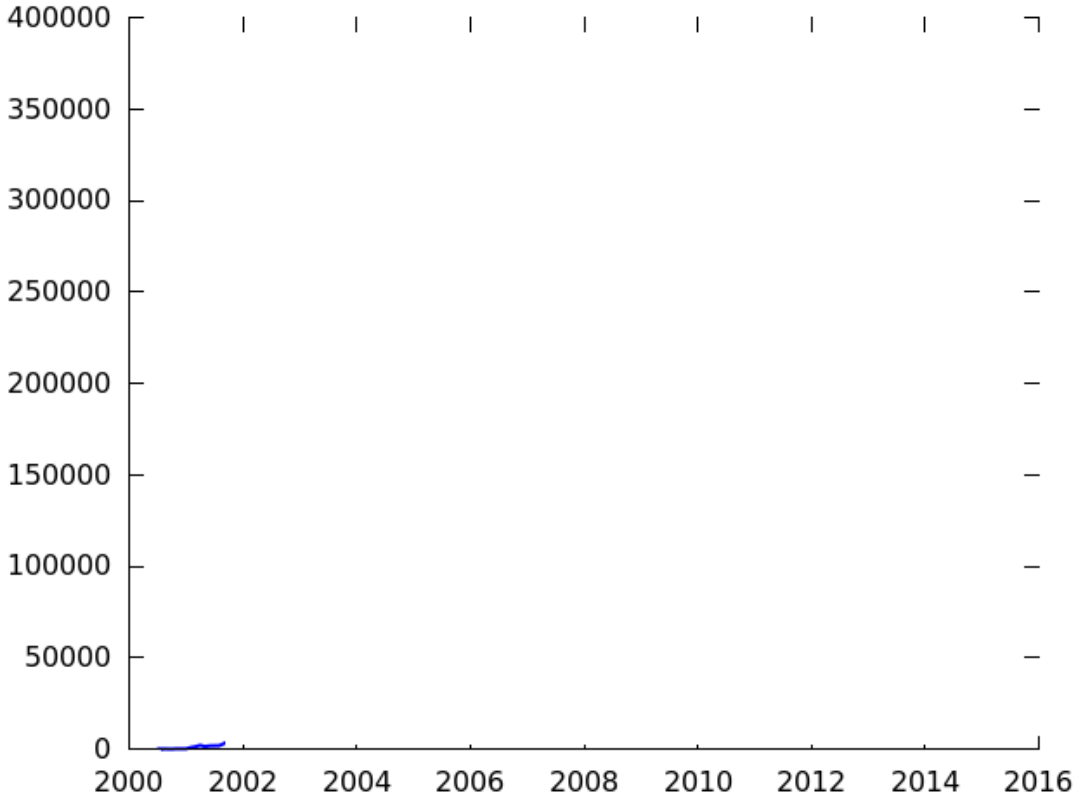
registered cppreference.com



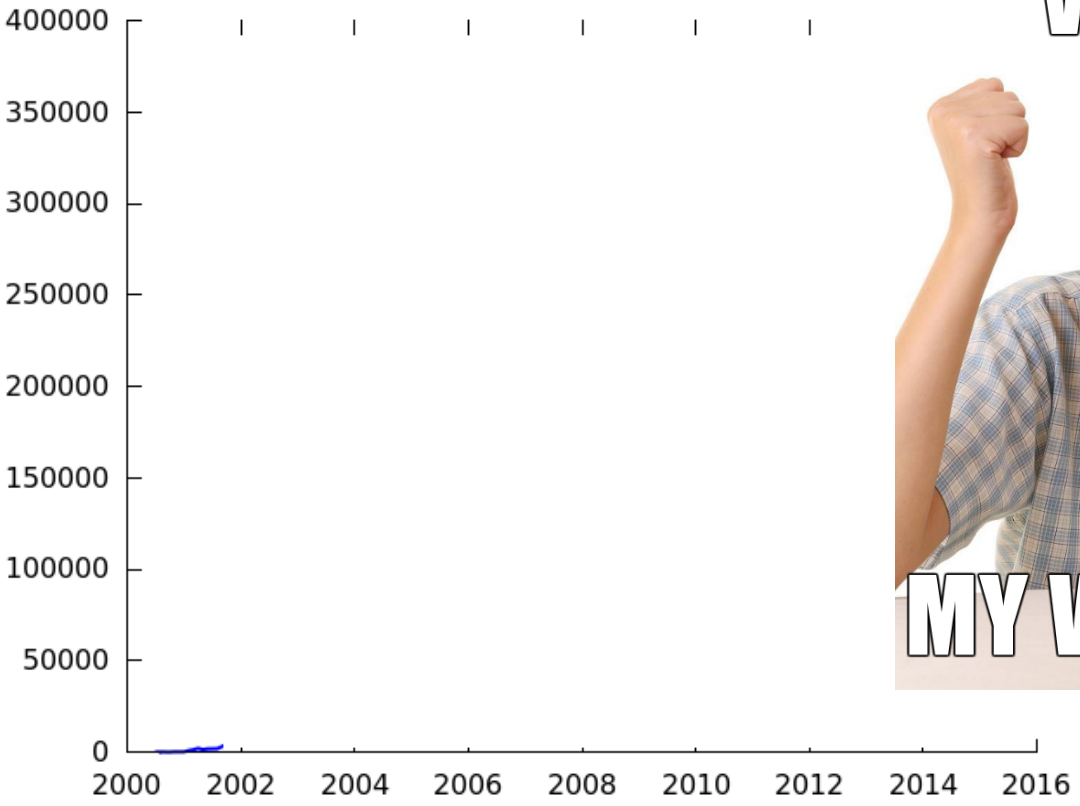
duplication is bad: size()

used a simple Perl templating system

First Steps


400000
350000
300000
250000
200000
150000
100000
50000
0
2000
2002
2004
2006
2008
2010
2012
2014
2016

First Steps


400000
350000
300000
250000
200000
150000
100000
50000
0
2000
2002
2004
2006
2008
2010
2012
2014
2016

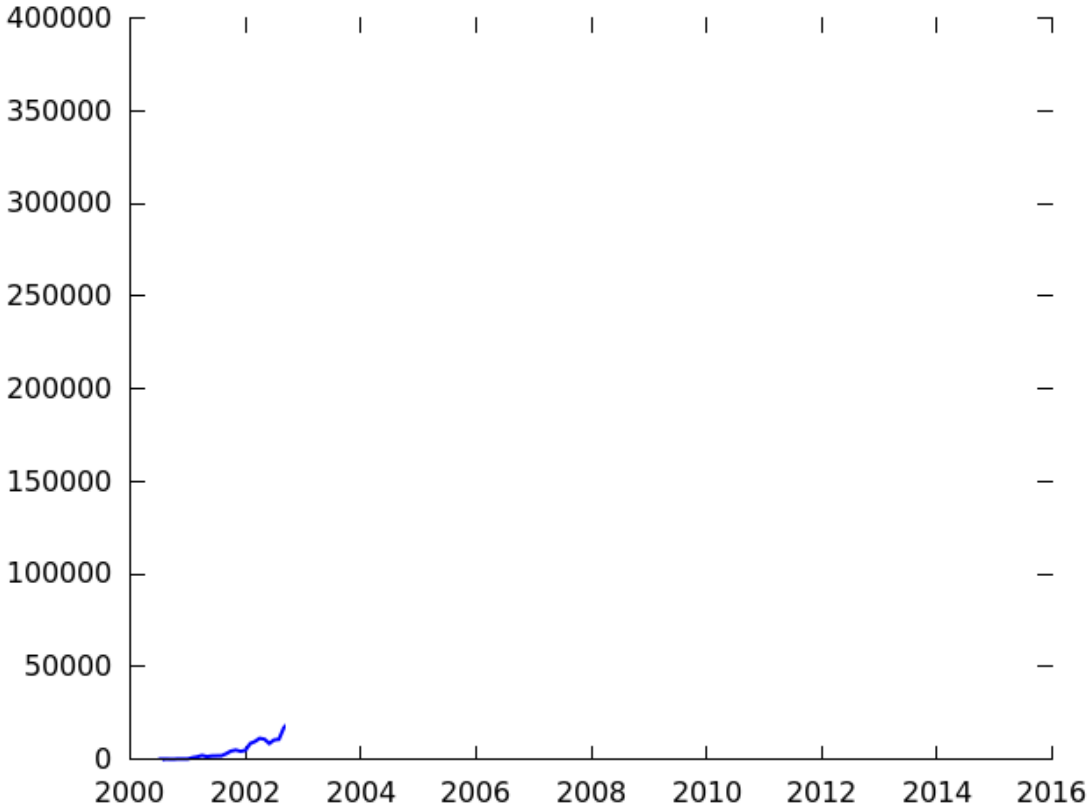

SOMEONE
VISITED
MY WEBSITE!

oh, Perl

“If you put a million monkeys at a million keyboards, one of them will eventually write a C++ program. The rest of them will write Perl programs.”

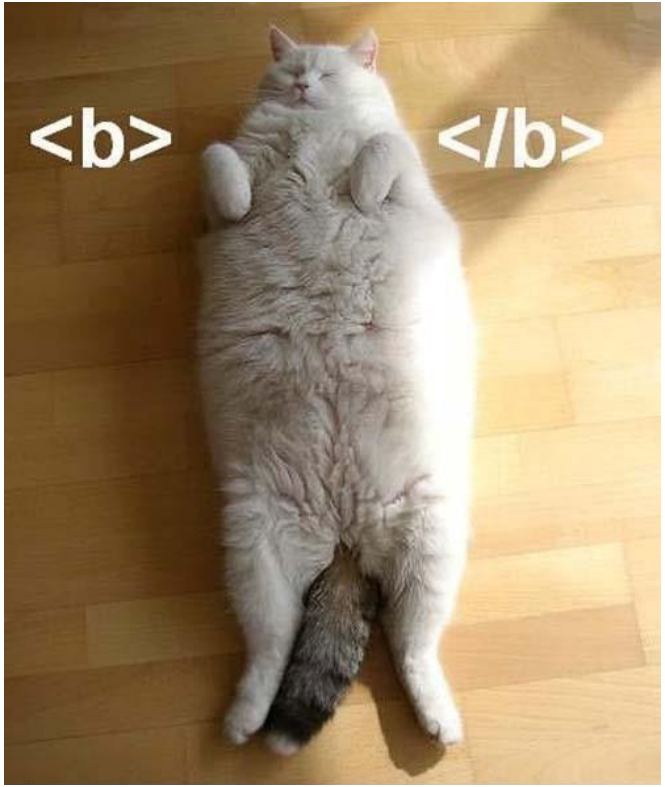
retire the templating system

The Site Grows



oh, HTML

HTML wasn’t great for
maintaining content
lots of repetition, structure
manually supporting IE6



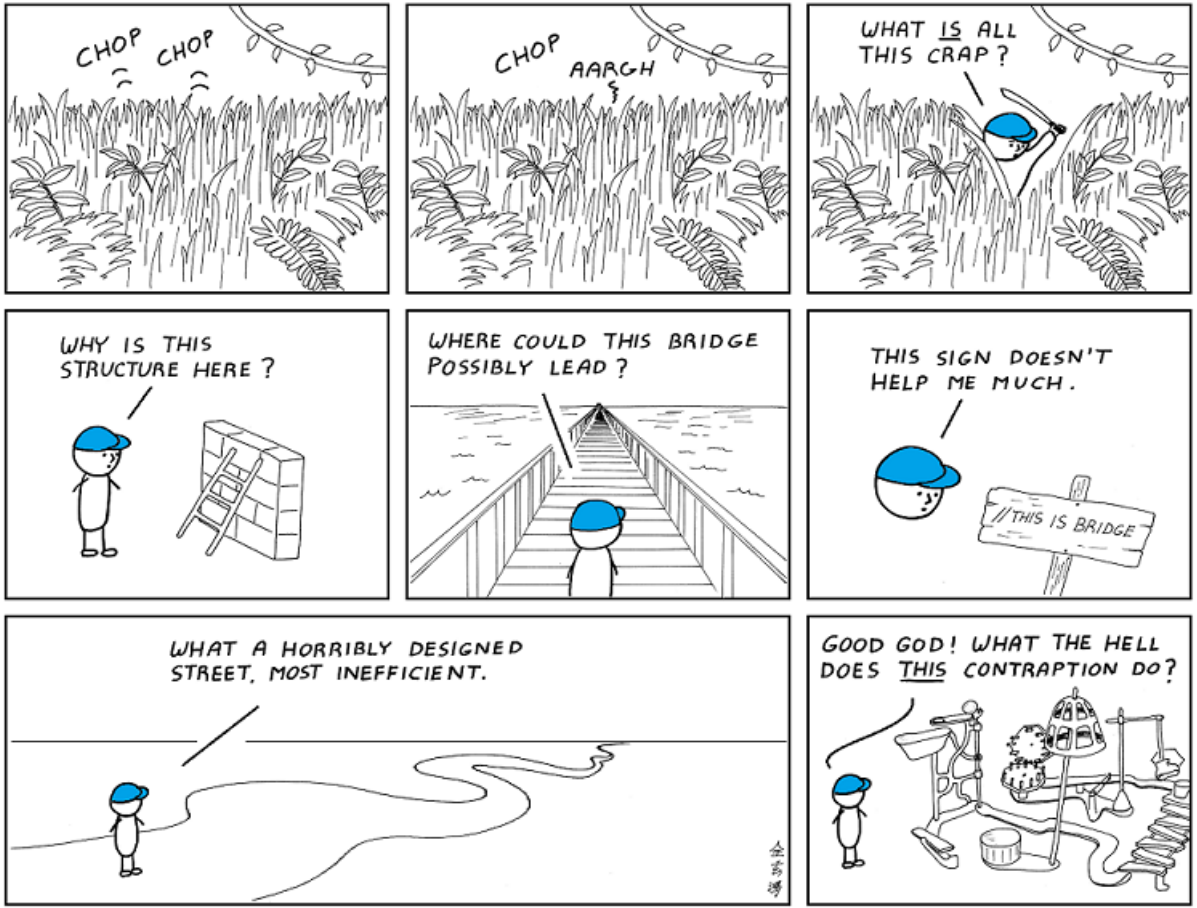
A Custom CMS

CMSes were all the rage (joomla, drupal)

php + mysql

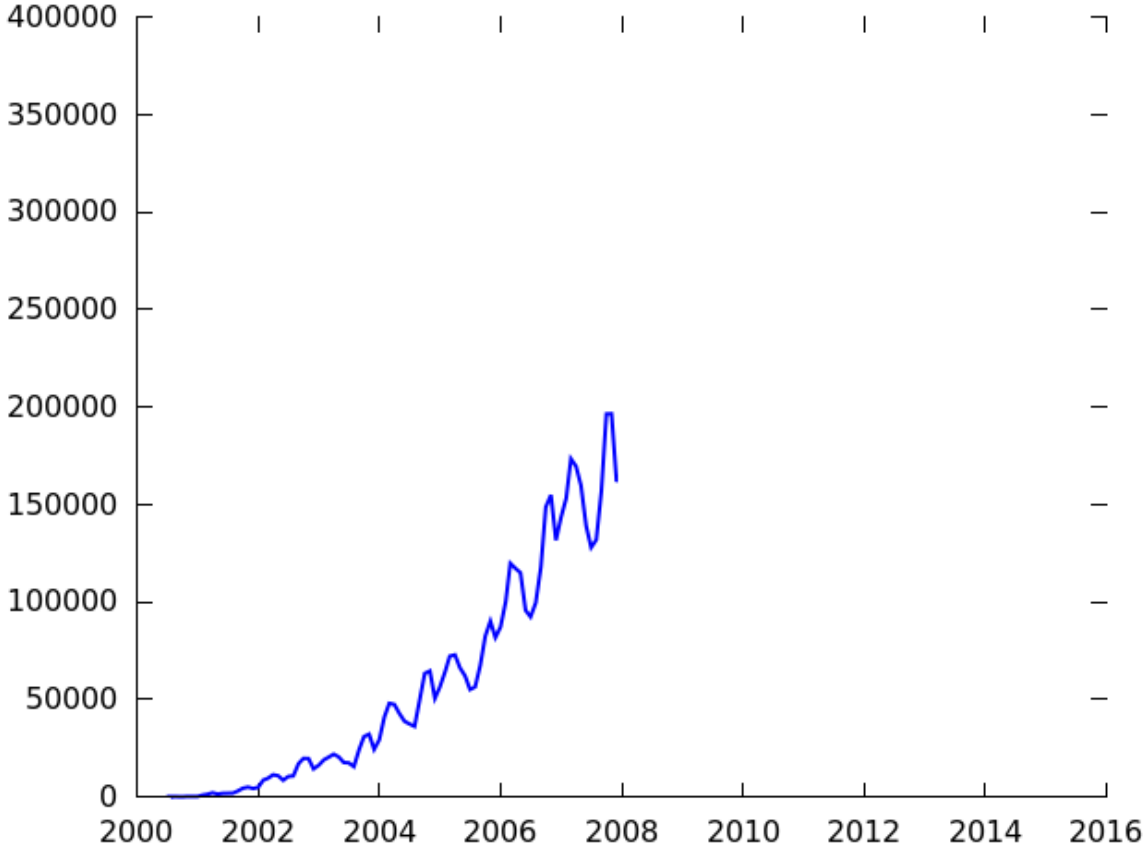
modularity again!

code my own, because...reasons



I hate reading other people's code.

The Site Continues to Grow

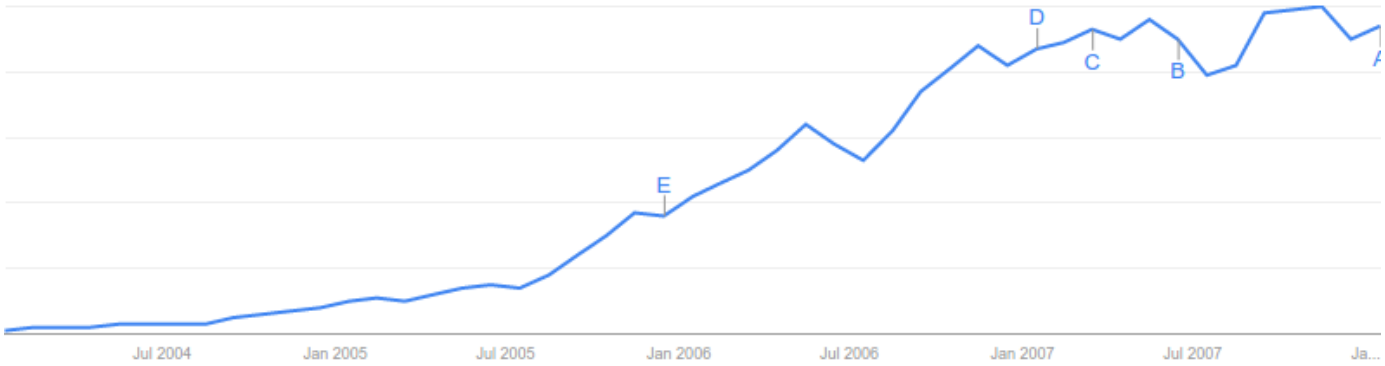


Crushing Guilt

Crushing Guilt

I’m a bottleneck

how to make it easier for others to contribute?



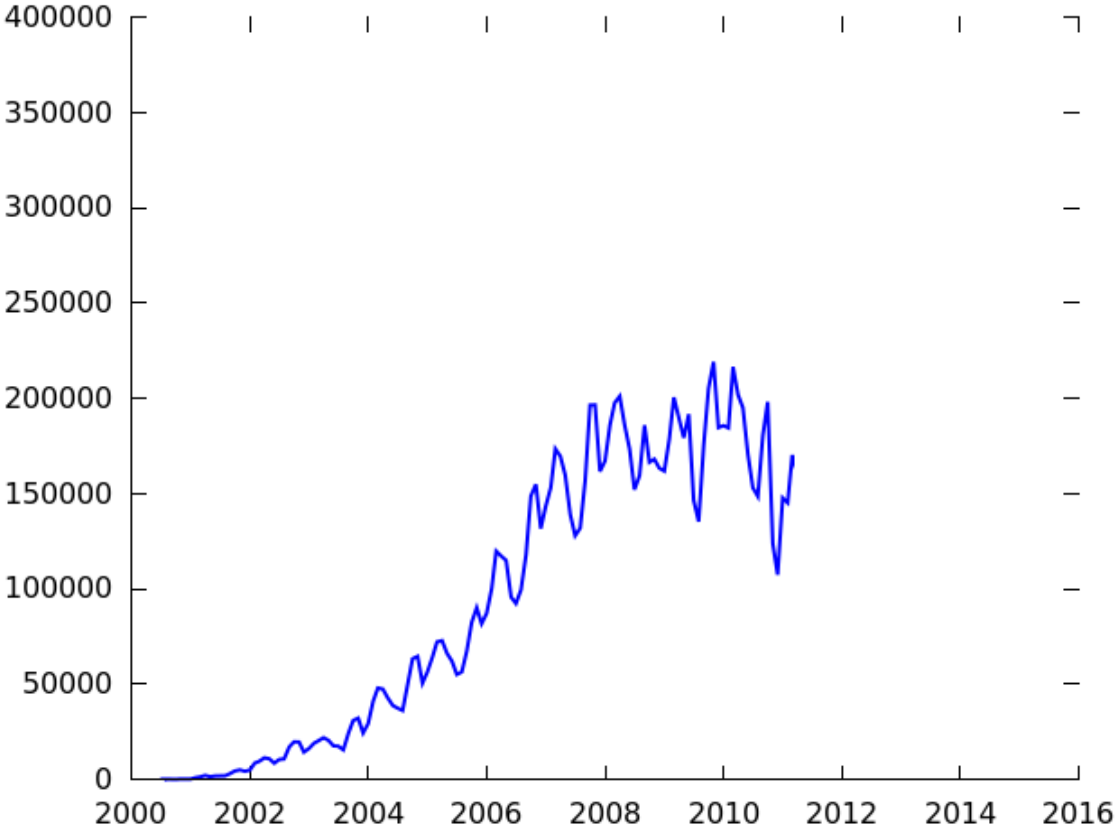
Make it a Wiki?

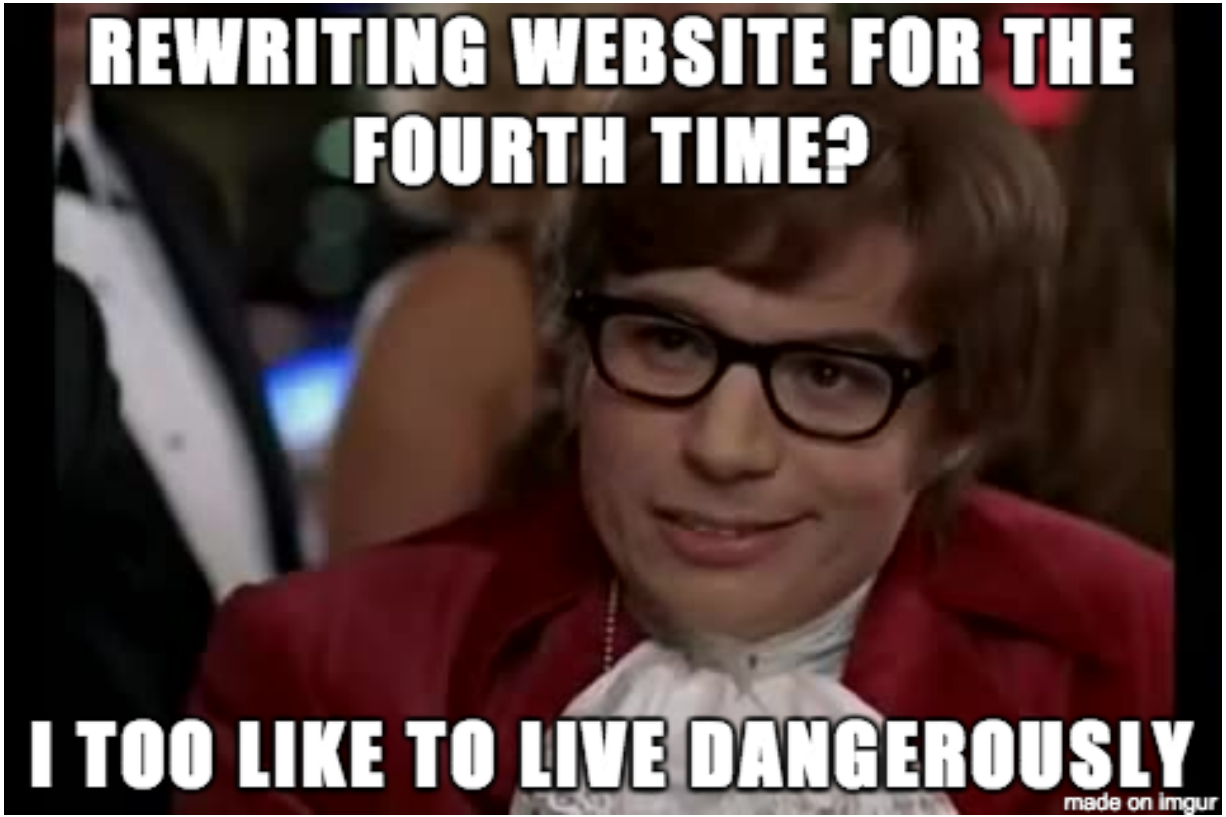
crowdsourcing

seemed to be working for wikipedia

dokuwiki: simple, easy, minimalist

The Site Continues to...Grow?




REWRITING WEBSITE FOR THE FOURTH TIME?
I TOO LIKE TO LIVE DANGEROUSLY
made on imgur

Need More Power

mediawiki has powerful templates

used by wikipedia

lots of expertise, plugins, etc.


```
template<class T>class L{L<T*>operator->()};L<int>i=i->
```

Need More Power

std::vector::size

```
size_type size() const;
```

Returns the number of elements in the container, i.e. `std::distance(begin(), end())`.

Parameters

(none)

Return value

The number of elements in the container.

Exceptions

(none)		(until C++11)
`noexcept` specification:	`noexcept`	(since C++11)

Complexity

Constant.

Example

The following code uses `size` to display the number of elements in a `std::vector<int>`: [edit]

Run this code

```
#include <vector>
#include <iostream>
```

Need More Power

```
{{cpp/container/{{{1|}}}/title | size}}
{{cpp/container/{{{1|}}}/navbar}}
{{dcl begin}}
{{dcl | since={{cpp/std|{{{1|}}}}} |
size_type size() const;
}}
{{dcl end}}

Returns the number of elements in the container, i.e. {{c|std::distance(begin(), end())}}.

===Parameters===
(none)

===Return value===
The number of elements in the container.

===Exceptions===
{{cpp/container/noexcept|{{{1|}}}}}

===Complexity===
Constant.

===Example===
{{include | cpp/container/{{{1|}}}/example_size}}
```

Need More Power

that page uses 60 templates

each of which can use more templates

complexity can be a problem

one of the first comments was “nice, but the templates are insane”

Complexity: Documentation

Template:dcl begin

This high-risk template has been protected from editing to prevent vandalism.
Please discuss any changes on the talk page.

Template documentation

This is one of the family of templates used for creation of declaration lists.

List template families

- dsc **** : For creation of member variable/function lists.
- dcl **** : For creation of detailed declaration lists (those including actual declaration code)
- sdsc **** : For creation of lists representing various syntaxes of a language feature. Used in subpages of cpp/language
- par **** : For creation of lists explaining function parameters.
- spar **** : For creation of lists explaining syntax parameters.
- nv **** : For creation of feature lists in navbars.

{{**dcl begin**}}

starts the declaration list

{{dcl end}}

ends the declaration list. This template automatically adds {{dcl sep}} above the item.

{{dcl rev begin| num=*num*}}

starts versioned declaration list. The num parameter is optional and may be used for numbering purposes (overrides the *num* parameter of the {{dcl}} templates used within the versioned list (i.e., until {{dcl rev end}}).

Complexity: Examples

Category:Todo no example

This category lists pages that have no examples, but should have one.

See also

- {{todo}}
- Category:Todo with reason
- Category:Todo without reason

Pages in category "Todo no example"

The following 200 pages are in this category, out of 418 total.
(previous 200) (next 200)

C

- c/io/fgetwc
- c/io/fputwc
- c/io/fwprintf
- c/io/fwscanf
- c/io/vfwprintf
- c/io/vfwscanf
- c/locale/lconv
- c/locale/setlocale
- c/numeric/complex/CMPLX
- c/string/byte/strxfrm
- c/string/multibyte/btowc
- c/string/multibyte/mblen
- c/string/multibyte/mbrlen
- c/string/multibyte/mbrtowc

C cont.

- cpp/container/unordered multiset/equal range
- cpp/container/unordered multiset/unordered multiset
- cpp/container/unordered set/emplace
- cpp/container/unordered set/equal range
- cpp/container/unordered set/unordered set
- cpp/container/vector bool/hash
- cpp/experimental/optional/hash
- cpp/experimental/optional/swap2
- cpp/io/basic filebuf/basic filebuf
- cpp/io/basic filebuf/close

C cont.

- cpp/io/basic streambuf/showmanyc
- cpp/io/basic streambuf/snextc
- cpp/io/basic streambuf/swap
- cpp/io/basic streambuf/uflow
- cpp/io/basic stringbuf/pbackfail
- cpp/io/basic stringbuf/swap2
- cpp/io/basic stringstream/operator=
- Template:cpp/io/basic stringstream/operator=
- cpp/io/basic stringstream/rdbuf
- Template:cpp/io/basic stringstream/rdbuf
- cpp/io/basic stringstream/swap
- Template:cpp/io/basic stringstream/swap
- cpp/io/basic stringstream/swap2

Balance

just like programming

finding a balance between repetition and reuse is important

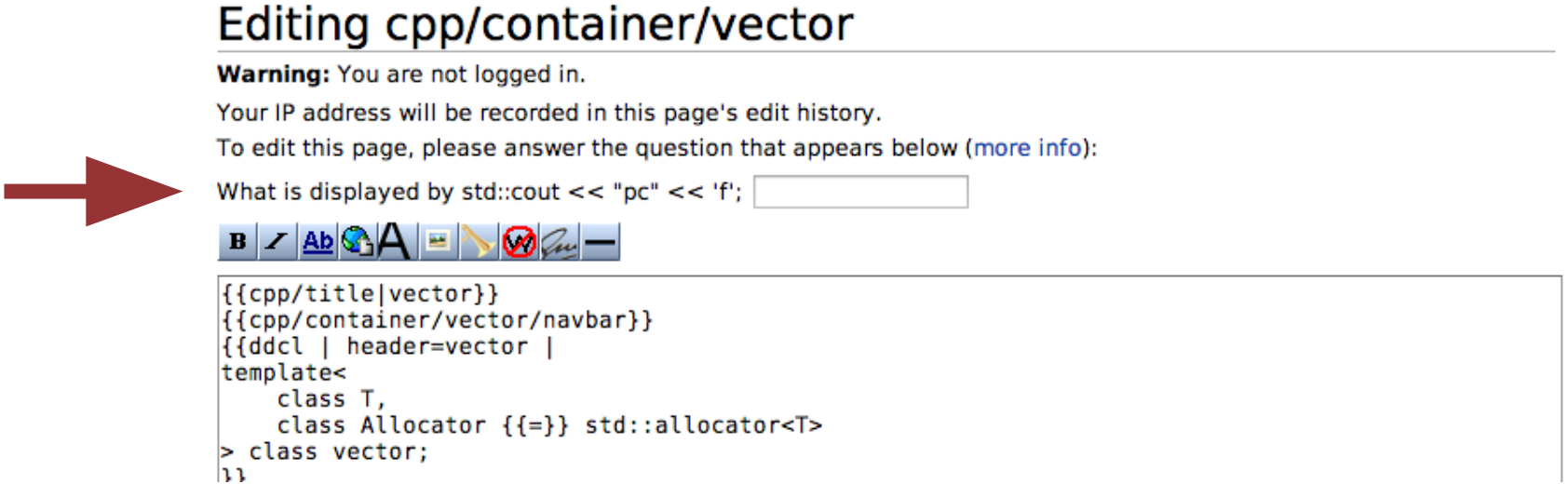
Vandalism

expected, but haven't got a lot?

helps to have ground truth

don't have contentious pages

Vandalism: CAPTCHAs



Editing cpp/container/vector

Warning: You are not logged in.

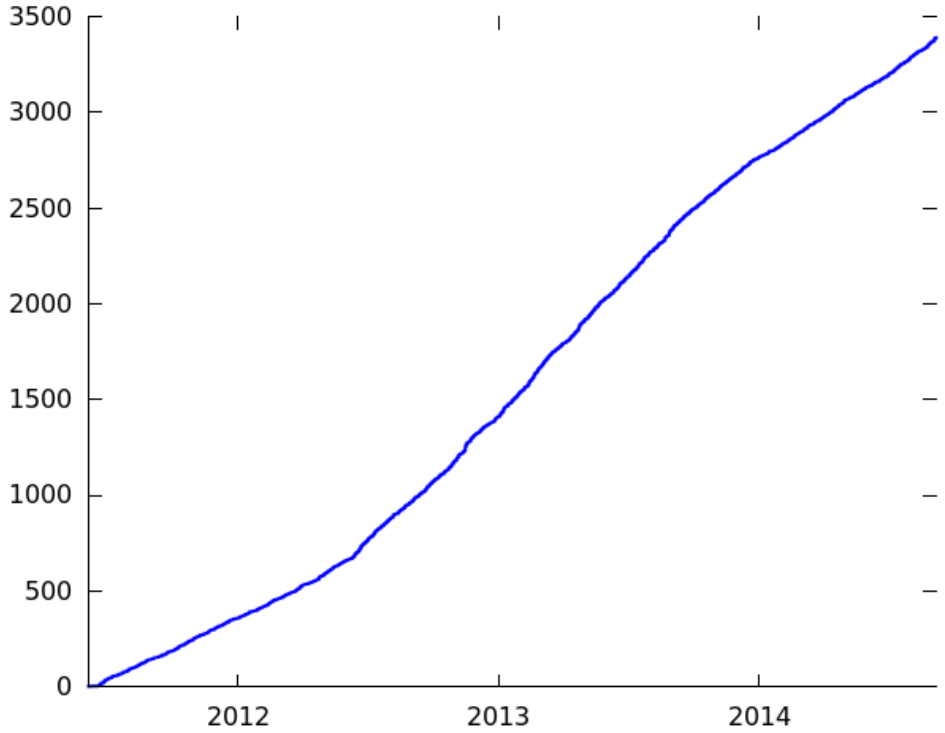
Your IP address will be recorded in this page's edit history.

To edit this page, please answer the question that appears below (more info):

What is displayed by std::cout << "pc" << 'f';

```
{{cpp/title|vector}}
{{cpp/container/vector/navbar}}
{{ddcl | header=vector |
template<
    class T,
    class Allocator {{=}} std::allocator<T>
> class vector;
}}
```

Vandalism - User Registrations?



Vandalism

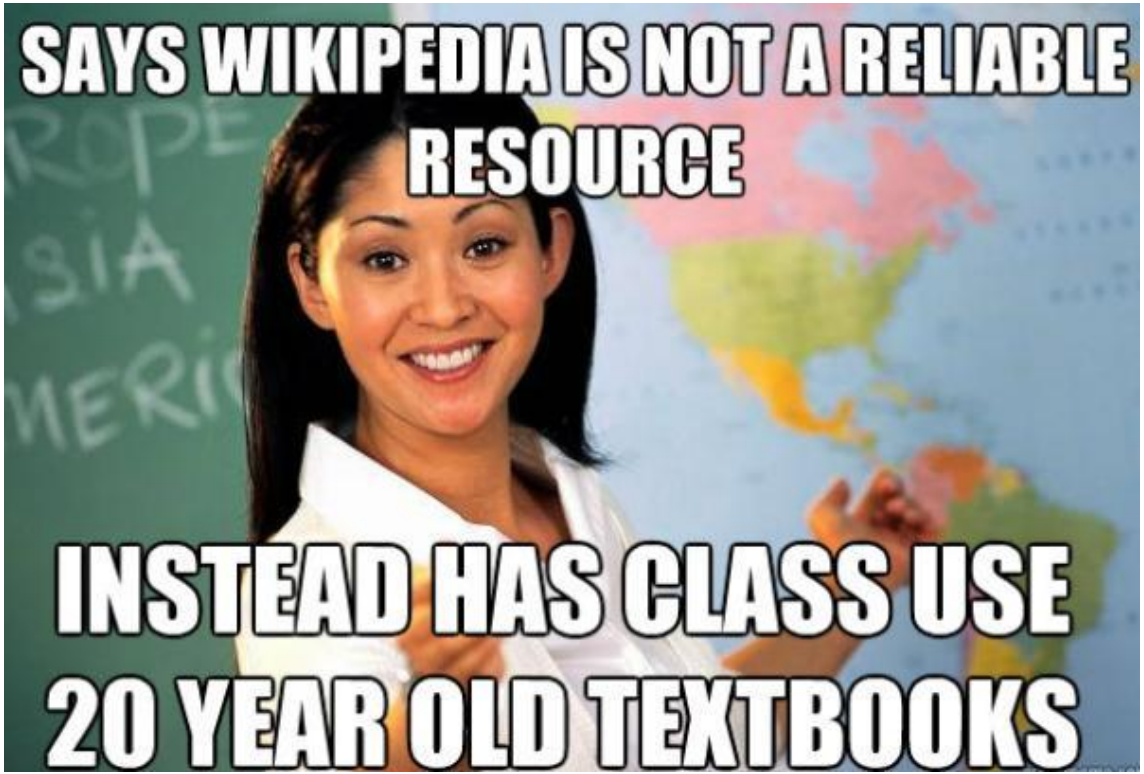
number of incidents: I can’t count that low

much more prevalent: “helpful vandalism”

- incorrect edits
- style fixups
- introducing new content that has problems

Trust

Trust


SAYS WIKIPEDIA IS NOT A RELIABLE RESOURCE
INSTEAD HAS CLASS USE 20 YEAR OLD TEXTBOOKS

The World of Tomorrow

new content

keep up with increased standard activity

flush out existing content (examples, TODOs)

translations, tutorials, boost?

features

The World of Tomorrow

Example

Run | Share | Exit | clang 3.4 (C++11)

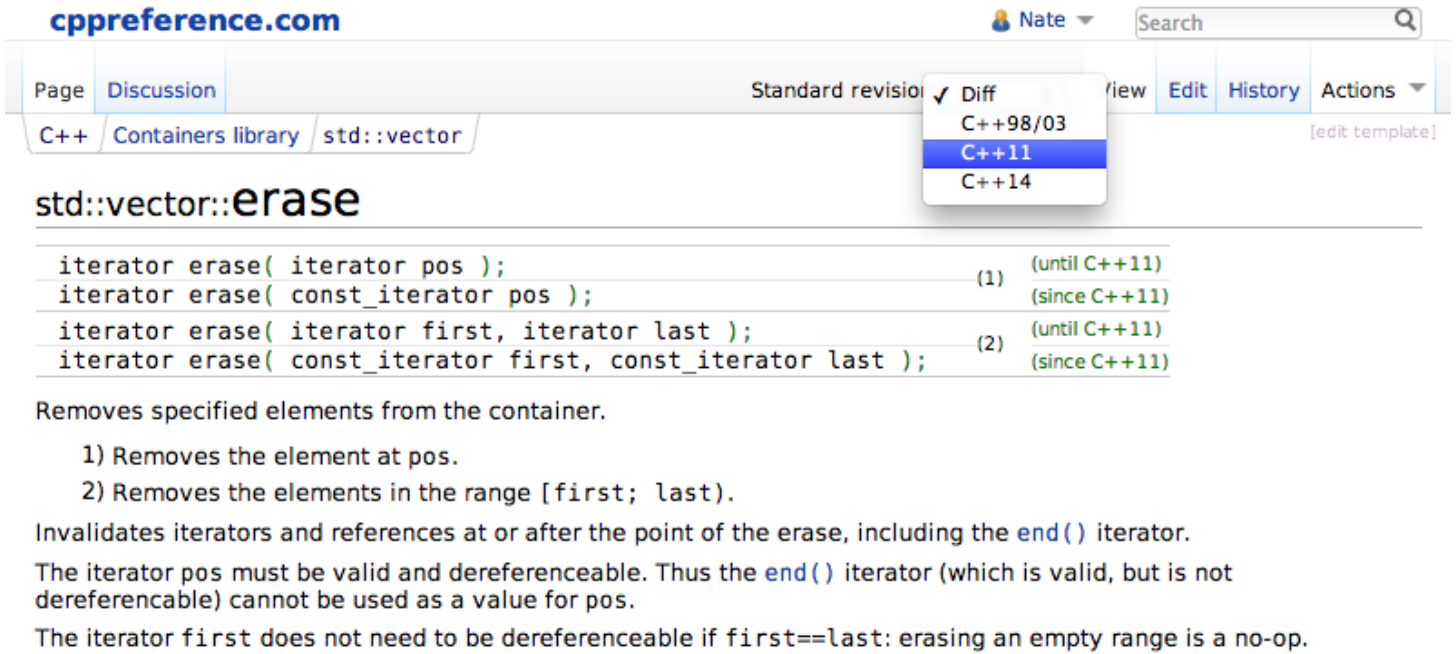
Powered by Coliru online compiler

```
1  #include <vector>
2  #include <iostream>
3
4
5  int main( )
6  {
7      std::vector<int> c{0, 1, 2, 3, 4, 5, 6, 7, 8, 9};
8      for (auto &i : c) {
9          std::cout << i << " ";
10     }
11     std::cout << '\n';
12
13     c.erase(c.begin());
14
15     for (auto &i : c) {
16         std::cout << i << " ";
17     }
18     std::cout << '\n';
19
20     c.erase(c.begin()+2, c.begin()+5);
21
22     for (auto &i : c) {
23         std::cout << i << " ";
24     }
```

Output:

```
0 1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9
1 2 6 7 8 9
```

The World of Tomorrow



Thank you!

...and thanks to all the contributors, especially P12 and Cubbi.