Chrome, Clang and C++

Hans Wennborg
hwennborg (at) google.com
7 March 2013

Outline

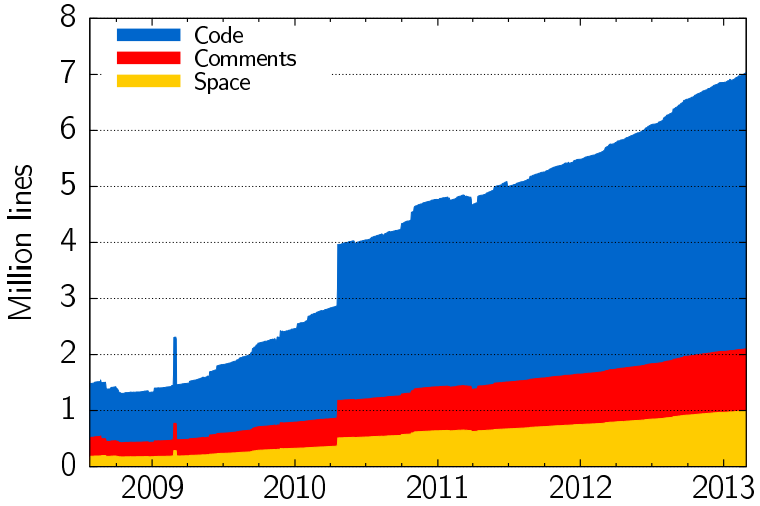
What is Chrome?

- A web browser from Google
- Pushing the web forward
- First released 2008
- 310 M active users (June 2012)
- Windows, Mac, Linux, ChromeOS, Android, iOS
- Mostly open source.

Chromium vs. Chrome

$$\text{Chromium} + \left\{ \begin{array}{l} \text{Branding} \\ \text{Quality assurance} \\ \text{Auto updates} \\ \text{PDF} \\ \text{Flash} \\ \text{Audio/Video codecs} \\ \text{Crash reports} \end{array} \right\} = \text{Chrome}$$

Lots of Code

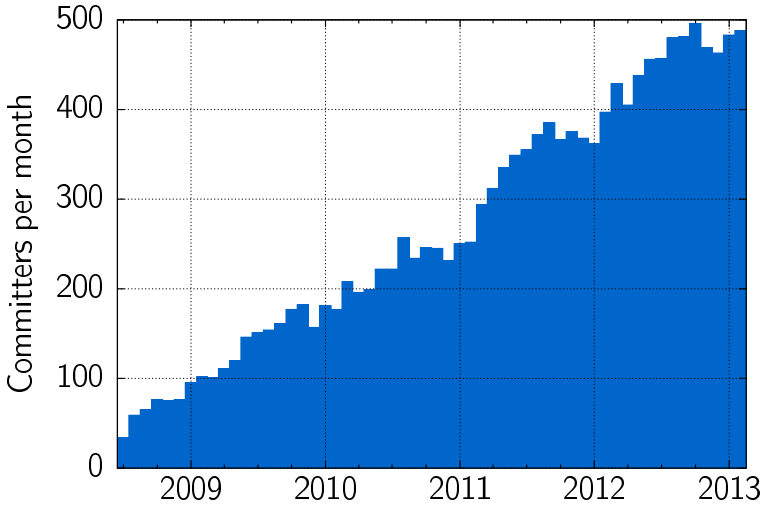


Lots of Code (continued)

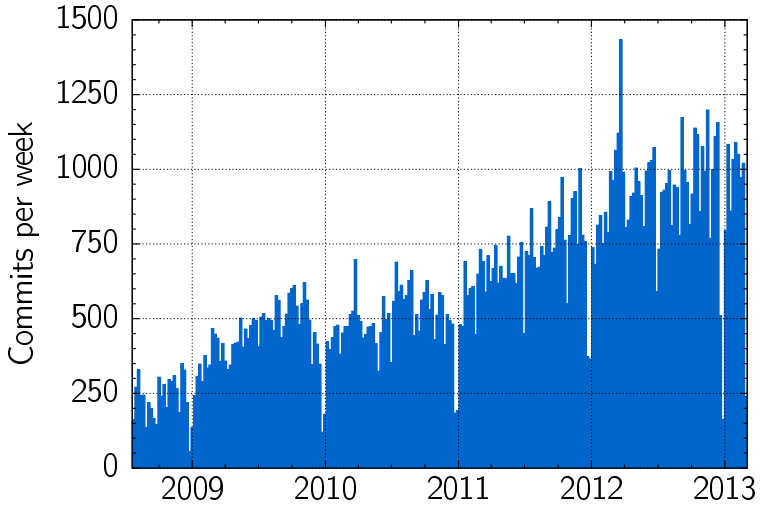
5 M lines more in third-party libraries:

- WebKit
- V8
- angle
- breakpad
- ffmpeg
- flac
- gmock
- googleurl
- gtest
- hunspell
- icu
- jsoncpp
- leveldb
- libjingle
- libsrtp
- libvpx
- lss
- NaCl
- nss
- sfntly
- skia
- webrtc
- . . .

Lots of People



Lots of Churn



Development Process

- Continuous builds
- Build sheriffs
- Try jobs and commit queue
- Code reviews.

Code Reviews

- Improves code quality
- Spreads knowledge
- Fosters good communication skills.

Clang

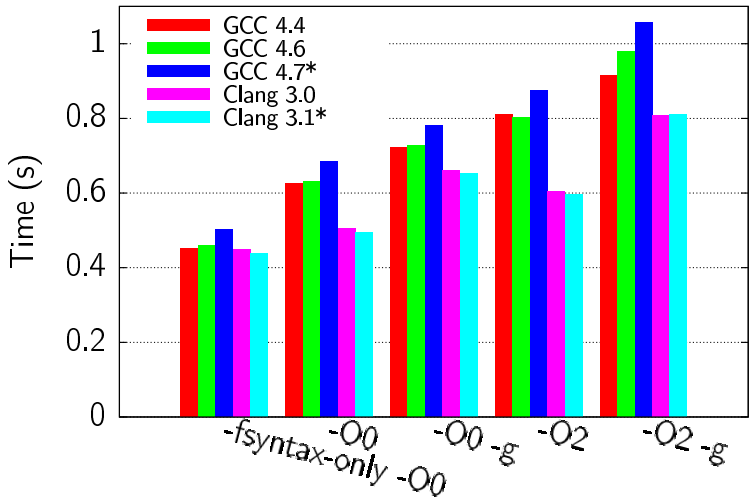
- Up-and-coming C++, C, Objective-C compiler
- Part of the LLVM project
- Announced 2007 by Apple
- Production quality compiler since ca 2010
- Open-source, BSD-style license
- Designed to be GCC compatible.

What makes Clang interesting?

- Speed
- Quality output
- Useful diagnostics
- Bug-finding warnings
- Hackable and extendable.

Compile Time

Random Chromium Files



What is a Compiler?

1. Translates code from one language to another
2. reporting any errors that it finds in the process.

The missing semicolon (GCC)

```
int f(int x) {
        int s = 0
        for (int i = 0; i < x; ++i)
                s += i;
        return s;
}
```

```
a.cc: In function 'int f(int)':
a.cc:3:9: error: expected ',' or ';' before 'for'
a.cc:3:25: error: 'i' was not declared in this scope
a.cc:3:35: error: expected ';' before ')' token
```

The missing semicolon (Clang)

```
int f(int x) {
        int s = 0
        for (int i = 0; i < x; ++i)
                s += i;
        return s;
}
```

```
a.cc:2:18: error: expected ';' at end of declaration
        int s = 0
                 ^
                 ;
1 error generated.
```

Typo correction

```
a.cc:5:9: error: use of undeclared identifier 'dout';
      did you mean 'cout'?
        dout << "Hello, world!" << endl;
        ^~~~
        cout
```

The Conditional Operator

```
int f(bool b, int x, int y) {
  return 7 + b ? x : y;
}
```

The Conditional Operator

```
a.cc:2:16: warning: operator '?:' has lower precedence
      than '+'; '+' will be evaluated first
  return 7 + b ? x : y;
         ~~~~~ ^
a.cc:2:16: note: place parentheses around the '?:'
      expression to evaluate it first
  return 7 + b ? x : y;
               ^
             (       )
1 warning generated.
```

The Boolean Accident

```
void f(bool* delete_user_data) {
        if (!migrate_data_to_new_location()) {
                delete_user_data = false;
                return;
        }

        ...
}
```

The Boolean Accident

```
a.cc:5:36: warning: initialization of pointer of type
    'bool *' to null from a constant boolean expression
              delete_user_data = false;
                                 ^~~~~
```

"In Case It's Sensitive"

md5.c:

```
/* In case it's sensitive */
memset(ctx, 0, sizeof(ctx));
```

"In Case It's Sensitive"

```
md5.c:6:31: warning: 'memset' call operates on objects
      of type 'context' while the size is based on a
              different type 'context *'
        memset(ctx, 0, sizeof(ctx));
               ~~~             ^~~
md5.c:6:31: note: did you mean to dereference the
      argument to 'sizeof' (and multiply it by the
      number of elements)?
        memset(ctx, 0, sizeof(ctx));
                               ^~~
```

Override

```
a.cc:2:22: error: 'foo' marked 'override' but does not
      override any member functions
        virtual void foo() override;
                     ^
```

Hackability

- Clang is extendable
- It is a set of libraries
- We can use it to build tools.

Chromium Style Checker

- Mark functions `virtual` explicitly
- Use `override` for overriding functions
- ...

Chromium Style Checker

```
In file included from a.cc:1:
./a.h:8:3: warning: [chromium-style] Overriding method
      must have "virtual" keyword.
  void foo();
  ^
1 warning generated.
```

`clang-format`

- Formatting is important
- Formatting is boring
- Automatic formatting of C++ is hard.

Live demo.

Summary

- Chrome is a large and active C++ project
- Tools are very important
- Clang is a new, fast, high-quality compiler
- With great diagnostics
- Allows new tools to be built on top of it.

References

- www.chromium.org
- clang.llvm.org
- clang.llvm.org/docs/ClangFormat.html