How to contribute to LLVM, Clang, etc
License

All codes are published under the « University of Illinois/NCSA Open Source License »
BSD-style

Some of them are also published under MIT licence (dual-license)

No copyright assignment
Some stats - LLVM

In a Nutshell, The LLVM Compiler Infrastructure...

... has had 103,599 commits made by 379 contributors representing 1,277,803 lines of code

... is mostly written in C++
   with an average number of source code comments

... has a well established, mature codebase
   maintained by a very large development team
   with increasing Y-O-Y commits

... took an estimated 362 years of effort (COCOMO model)
   starting with its first commit in June, 2001
   ending with its most recent commit about 4 hours ago

30 Day Summary
Jan 1 2014 — Jan 31 2014
1108 commits
99 Contributors
   including 5 new contributors

12 Month Summary
Jan 31 2013 — Jan 31 2014
12410 commits
   Up +1837 (17%) from previous 12 months
236 Contributors
   Up +57 (31%) from previous 12 months

February 2nd, 2014
How to contribute to LLVM
Sylvestre Ledru

Source: ohloh.net
Some stats - Clang

In a Nutshell, LLVM/Clang C family frontend...

- has had 49,680 commits made by 273 contributors representing 827,471 lines of code
- is mostly written in C++
  with an average number of source code comments
- has a well established, mature codebase
  maintained by a very large development team
  with stable Y-O-Y commits
- took an estimated 230 years of effort (COCOMO model)
  starting with its first commit in July, 2007
  ending with its most recent commit about 13 hours ago

30 Day Summary
Jan 1 2014 — Jan 31 2014

603 Commits
70 Contributors
including 3 new contributors

12 Month Summary
Jan 31 2013 — Jan 31 2014

7494 Commits
- Down -390 (4%) from previous 12 months

176 Contributors
- Up +34 (23%) from previous 12 months

February 2nd, 2014

How to contribute to LLVM
Sylvestre Ledru

Source: ohloh.net
Community

Friendly
Mainly professionnal (Apple, Google, ARM, Linaro, Intel, etc)
With also individual and academic
Usually fast to answer to comments/questions
Mailing lists

Plenty of them (and high traffic!)

**LLVM**
- LLVM-dev
- LLVM-commits

**Clang**
- cfe-users
- cfe-dev
- Cfe-commits

Other mailing lists for other LLVM projects
Repositories

Using subversion
git mirrors available

Permissions are granted on all LLVM projects (clang, compiler-rt, lldb, polly, etc) and even the websites
Write a patch?

Make the patch against the current trunks (!)

Made with svn diff (git diff or diff -u)

Must have tests or explain why testing is not possible

Must pass the whole test suite

When relevant, update the release notes
Review process

Documented on
http://llvm.org/docs/DeveloperPolicy.html

Minor patches (typo, trivial bug fix, etc) can be committed directly without review

For non-regular contributors, {llvm,cfe,lldb}-commits can be used

No private review
Contributing a patch

• Send to llvm-commits, clang-commits

• Wait for reviewer

• Address comments

• Wait for 'LGFM' (Looks good for me)

• Commit patch
  - if you have commit rights -> commit yourself
  - if not -> ask for the patch to be committed
Some advices

- Make the patch as small as possible
- One patch <=> One feature
- Extract unrelated (trivial) fixes into separate patches
- CC possible reviewers
Some advices (bis)

- Pinging patches:
  - Sometimes patches slip through
  - Ping them after 5-7 days (or holidays)
  - Include the latest patch, rebased to 'trunk' in the ping mail

- Make yourself known
Review patches

- Do not need to be a code owner to review patches.

- Reviewers, even if they can not 'LGTM' patches, are highly welcome.

- If you review patches, people are more likely to review your patches.

- Especially review patches in areas you contributed to before.
Large patches / projects

• Discuss the design on the mailing list, before starting the development

• Features are developed, reviewed and committed incrementally

• Make most patches 'trivial'

• Do by no means _develop_ a large patch in the dark. During the review, you will basically write the feature a second time
Arcanist + phabricator

An other review system!

Phabricator:
A code review platform. Hosted on:
http://llvm-reviews.chandlerc.com/
Arcanist + phabricator

February 2nd, 2014
How to contribute to LLVM
Sylvestre Ledru
February 2nd, 2014

How to contribute to LLVM

Sylvestre Ledru
Arcanist + phabricator

Arcanist:
CLI tools build to interact with phabricator
Works on top of git/mercurial
Get +w permissions

After a few accepted patches
Super easy + fast

Username +
'fullname <email>' +
password hash
To Chris Lattner

Example: Request sent at 15:15, account opened at 22:30
Coding standards

http://llvm.org/docs/CodingStandards.html

Defines headers, methods naming, C++ usage (example : no RTTI/exception), etc
Open Projects

LLVM :
http://llvm.org/OpenProjects.html

Clang :
http://clang.llvm.org/OpenProjects.html

After that, we have a few other projects:
Polly (see later today), lldb, lld, compiler-rt, libc++, etc
• Fix some bugs: http://llvm.org/bugs/

• Fix some (easy) issues found by the static analyzer: http://buildd-clang.debian.net/scan-build/
Questions ?