Finally Building on Windows

Stephan Bergmann

FOSDEM, January 2016

Nothing frightens me more
Than religion at my door

—John Cale

Nothing frightens me more
Than religion at my door

—John Cale

Except, maybe,
Microsoft Visual Studio
on my screen

Monoculture

- Generic parts of LO built in lots of settings
  - GCC, Clang, MSVC
  - Different sets of warnings, plugins
  - Dynamic sanitizers

    healthy

- Windows-only parts are only built with MSVC

    unhealthy

clang-cl to the rescue

- Clang with an MSVC-style frontend
  - Understands MSVC's command line options
  - Understands MSVC's language extensions
    - __cdecl, __declspec(dllexport), etc.
- Sufficiently mature in Clang 3.8/trunk
- “Compiling large, real-world codebases with clang on Windows” by Hans Wennborg, Nico Weber

(Only need to run Visual Studio once to build Clang on Windows)

Setup

- CXX=.../clang-cl.exe -D_CRT_RAND_S= -FIIntrin.h -fmsc-version=1800 -Qunused-arguments --target=x86_64-pc-windows-msvc
  - Some MSVC intrinsics (__stosb) come from clang-cl Intrin.h instead
  - sal/osl/w32/random.c: #define _CRT_RAND_S, #include <stdlib.h>
  - -Qunused-arguments: -GS ("buffer security check"), -Zc:wchar_t-
- --disable-activex
  - see https://llvm.org/bugs/show_bug.cgi?id=13737#c5 (also, some configure.ac ATL_INCLUDE confusion)
- --disable-pch, --disable-compiler-plugins
- http://people.redhat.com/sbergman/0001-clang-cl-no-climaker.patch

Proudly breaking toolchains...

- For MS ABI, emit dllexport friend functions defined inline in class
  - struct S { friend __declspec(dllexport) void f() {} };
- clang-cl: Take dllexport from original function decl into account
  - struct __declspec(dllexport) Outer {
      void f(); struct Inner { friend Outer::f(); };
    };
- clang-cl: support __cdecl-on-struct anachronism
  - ICU's gendict generates “struct {...} __cdecl s;”
  - MSVC: “warning C4229: anachronism used : modifiers on data are ignored”

Proudly breaking toolchains...

- clang-cl: vtordisp thunks not emitted for functions with class template specializations in their signatures
  - http://people.redhat.com/sbergman/0002-TODO-work-around-clang-cl-ABI-bug-PR25641.patch
- workdir/UnpackedTarball/icu/source/tools/toolutil/Makefile:
  - $(CC) ... "-DU_HOST=\"x86_64-unknown-cygwin\"" ...
  - error: expected expression: ... U_HOST ...
    expanded from command line:
      #define U_HOST \\\x86_64-unknown-cygwin\\\\

(At least, when you run into a clang-cl ICE, you can debug it on Linux)

Bugs found

- -Wint-to-pointer-cast: Stuffing 64-bit pointer values into 32-bit integers
  - (“long” is only 32-bit in MSVC 64-bit mode)
- -Wbitwise-op-parentheses
  - if ((m_nStyle & dottedLine|solidLine) != 0)
- -Wunused-private-field

- Alas, cannot run our Clang plugins
  - Patch on the Net changing plugin registration to work in principle, but requires to build clang-cl with shared-library support, which seems broken

Miscellanea

- Order of (non-standard) attributes:
  - class SAL_WARN_UNUSED BASEGFX_DLLPUBLIC B3DTuple …
- “#pragma warning (push, 1)” effectively ignored
- Restricted debug info (file/line info, but no variables)

THANK YOU